FORMATION CONTROL AND PATH PLANNING STRATEGIES FOR

UNMANNED AERIAL VEHICLE SWARMS

Srijita Mukherjee

Dissertation Prepared for the Degree of

DOCTOR OF PHILOSOPHY

UNIVERSITY OF NORTH TEXAS

August 2021

APPROVED:

Kamesh Namuduri, Major Professor
Murali Varanasi, Committee Member
Bill Buckles, Committee Member
Gayatri Mehta, Committee Member
Amjad Soomro, Committee Member
Shengli Fu, Chair of the Department of Electrical Engineering
Hanchen Huang, Dean of the College of Engineering
Victor Prybutok, Dean of the Toulouse Graduate School

Mukherjee, Srijita. *Formation Control and Path Planning Strategies for Unmanned Aerial Vehicle Swarms*. Doctor of Philosophy (Electrical Engineering), August 2021, 79 pp., 33 figures, 86 numbered references.

This dissertation focuses on the path planning of unmanned aerial vehicle (UAV) swarms under distributed and hybrid control scenarios. It presents two such models and analyzes them both from theory and practice. In the first method, a distributed formation control strategy for UAV swarm based on consensus law is presented. This model makes use of the fundamental concepts of leader-follower structure, social potential functions, and algebraic graph theory to jointly address flocking and de-confliction in the formation control problem. The impact of network topology on formation control is analyzed. It is shown that the degree distribution of the network representing the multi-agent system defines the rate at which formation is attained. Conditions for convergence and stability are derived. In the second method, a hybrid framework for path planning and coverage area by UAV swarms is presented. This strategy significantly improves the current labor-intensive and resource-constraint operations in aquaculture farms. To monitor the farms periodically, an optimized back-and-forth flight path based on the Shamos algorithm is utilized. A trajectory tracking strategy for UAV swarms under uncertain wind conditions is presented.

ACKNOWLEDGEMENTS

I take this opportunity to praise and thank the Almighty for all the shower of blessings for the successful completion of my doctoral research at UNT.

I express my sincere gratitude to my thesis advisor, Dr. Kamesh Namuduri, for providing invaluable guidance and support throughout my research. It was his guidance that supported me every step of the way and always seek the right questions. I thank him for all his suggestions for my research.

I am grateful to my committee members, Dr. Murali Varanasi, Dr. Bill Buckles, Dr. Gayatri Mehta, and Dr. Amjad Soomro for the valuable advice and support during my Ph.D. I would like to thank all the faculty members and staff from the Department of Electrical Engineering, the University of North Texas for their support and encouragement.

Finally, I am sincerely grateful to my parents, Dr. C.K. Mukhopadhyay and Baisakhi Mukhopadhyay for encouraging me to pursue my research work and for being constant pillars of support for the completion of my Ph.D. at UNT.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1. Motivation

Autonomy plays an important role in simplifying the engineering processes. The development of an autonomous system stems from the idea of adding intelligence. Adding intelligence enables a system adapt to dynamic conditions without human supervision. Autonomous systems created a great impact in today's world allowing complex operations to be performed efficiently. A network of such autonomous systems is termed a Multi-Agent System (MAS). A MAS is a combination of individual systems that work together to accomplish a set of tasks. They can be composed of both homogeneous and heterogeneous systems. There are numerous applications of MAS in robotics, defense, assembly line in the industries, and many more. The Perseverance Rover sent to Mars by NASA is a potent example of reliability, effectiveness, intelligence, and precision obtained by an autonomous system.

The study of a MAS is complimentary to the multi-disciplinary nature of its design. Various fields of study such as graph theory, control theory, communication networks, information technology, cyber-security, robotics, and other mathematical concepts are used to build and analyze a MAS. The MAS is designed using both centralized and distributed systems of control. Cooperative control is defined as the distributed control of a MAS where decisions are taken unanimously by all the agents. One of the popular applications of cooperative control is the Formation Control of MAS. It is a grouping technique adopted by military air-crafts during World War. The formation of aircraft in shapes like Vee, Diamond, Line, etc, proves useful in the reduction of drag and improving fuel efficiency. It also serves as camouflage for the air-crafts during wars. Thus, formation control involves the convergence of several autonomous agents to a desired geometric pattern. Formation control inherently plans the path of the agents, using established control laws while converging to the geometric pattern. So, it can be said that path planning is a fundamental process in cooperative control that provides way-points for the agents to traverse.

In this Dissertation, we focus on two major applications of multi-agent systems - Formation Control and Motion Planning of MAS.

1.2. Dissertation Contributions

1.2.1. Formation Control of UAVs

Innovations in the field of Science and Engineering have always been inspired by natural phenomena. The collective behavior observed in the flocks of birds, swarms of bees, schools of fish, colonies of ants, etc, has influenced the development of aspects addressing the coordination in MAS. Cooperative control is a problem of controlling multiple agents that interact with one another and take decisions to fulfill individual (local) and common (global) objectives. It is one of the branches of distributed control. In cooperative control, information sharing among the agents plays an important role. The information exchange between multiple agents in a network is called "Consensus". It is an agreement protocol that enables all agents to achieve convergence to a common value. Consensus protocols are used in several applications of swarming, flocking, rendezvous, and formation control.

When multiple agents arrange themselves in a specific geometry to perform a task, it is defined as "Formation Control" as shown in Fig 1.1. The formation of unmanned vehicles is deployed in both civil and military spheres for surveillance, fire control, underwater exploration, and rescue operations.

One of the approaches to distributed formation control is the use of consensus-based controllers. Formation control also requires the agents to maintain a geometric pattern while converging to a desired state. Maintenance of a desired geometric configuration leads to enhanced feasibility, accuracy, robustness, flexibility, lower cost, energy efficiency, and probability of success. Applications of formations include terrain model acquisition, measurement of radiation levels, planetary exploration, defense, etc. In space applications, multiple small spacecraft are also used in place of a single larger spacecraft to reduce mission cost and improve robustness and accuracy [2].

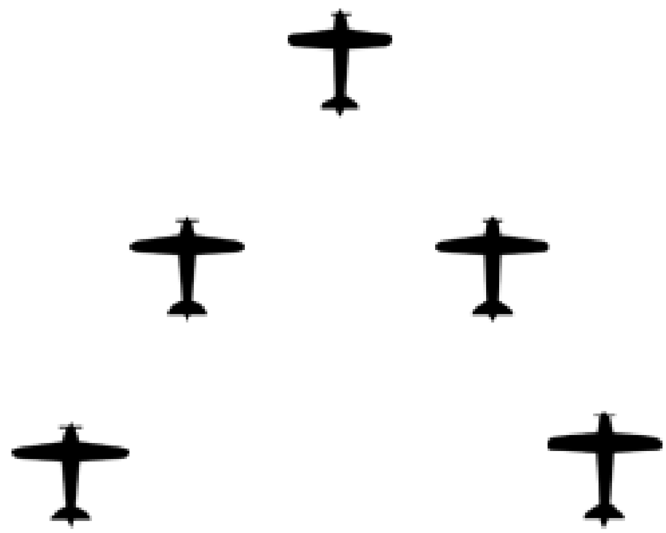

FIGURE 1.1. V formation

The interaction of multiple agents in a network is one of the building blocks of formation control. This makes the "interaction topology" or "graph topology" of the network of MAS central to the process of system analysis. The interaction topology of a MAS is typically represented by a graph. Here, each agent acts as a node of the graph and the connections between the agents form the links or edged between them. The operations on the graph that represents a MAS are governed by the principles of "Graph Theory".

In this work, we propose a solution to distributed formation control problem using consensus-based laws along with collision avoidance. We do this by classifying the area around an agent into "Collision Zone" and "Interaction Zone". The application of repulsive force generated by social potential functions helps in collision avoidance among the agents during the formation process.

Also, the effects of the degree distribution of the MAS interaction graph on the formation control process are studied. It is observed that the variation in the number of degrees per node affects the convergence to formation.

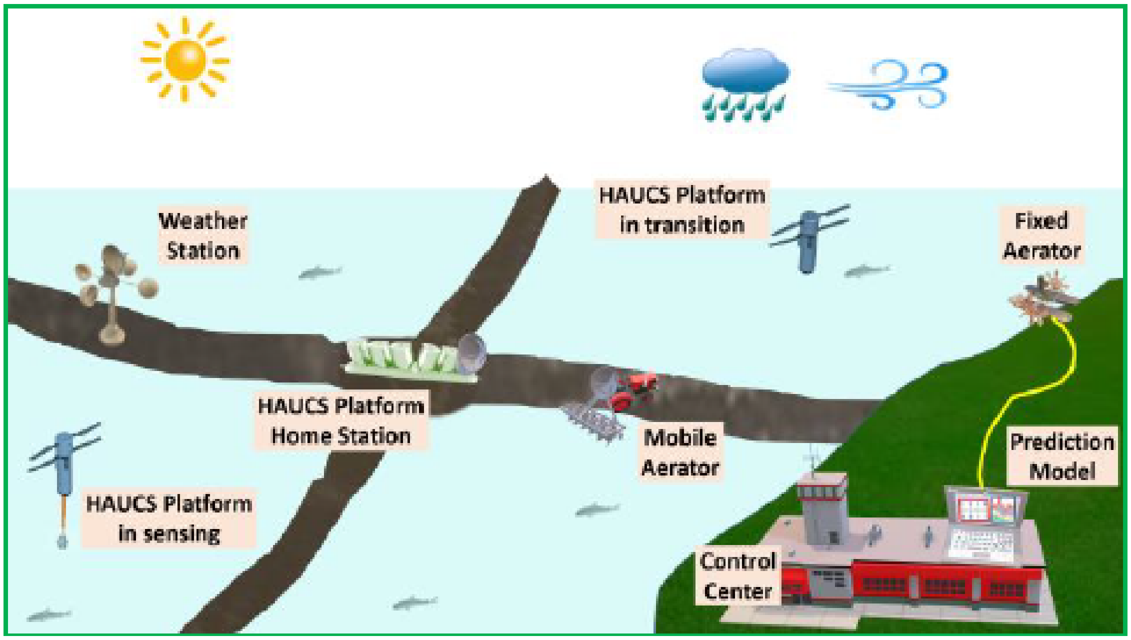

FIGURE 1.2. HAUCS Framework [1]

1.2.2. Path Planning of UAVs

Unmanned aerial vehicles (UAVs) form an important aspect of the "Internet of Things (IoT)" framework. One of the applications of the IoT framework is "Precision Agriculture (PA)". It involves the usage of robotic technologies in agriculture to improve farm production. Recently, there has been an increase in the usage of robotic technologies in PA [3, 4, 5]. One of the fast-growing sectors of agriculture is aquaculture farming, which has seen many implementations of robotic technologies and IoT [6, 7, 8, 9].

HAUCS stands for Hybrid Aerial/Underwater RobotiC System and consists of an unmanned robotic vehicle along with submerged underwater sensors. It is an IoT framework developed in [1] that involves a collaboration between multiple unmanned/robotic platforms, central operator, and machines for monitoring aquaculture farms. The HAUCS framework is shown in Fig 1.2.

HAUCS autonomous unmanned platform (AUP) is a robotic platform that can fly in the air and also traverse through the water surface. Monitoring of critical parameters such as "Dissolved Oxygen (DO)" in the pond waters is very important for the successful operation of the pond-based fish farms. The sampling and monitoring of DO by human operators is a labor-intensive and costly procedure. The introduction of HAUCS AUPs in fish farms offers the following advantages:

(1) Improved Scale of Operation: The HAUCS platforms can be deployed on farms of various scale;

(2) Accurate Measurements: Sampling of multiple locations on the farm offers an accurate measure of DO level; and

(3) Reduced Bio-fouling: The permanent placement of sensors is not required due to quick measurements by HAUCS.

Path planning for the periodic monitoring of fish farms by the AUPs is the central aspect of the HAUCS framework. A hybrid control strategy consisting of:

(1) a Centralized path allocation to HAUCS autonomous unmanned platforms (AUP); and

(2) an Autonomous trajectory tracking of AUPs under wind disturbances. is proposed.

The optimal coverage path planning is based on the Shamos Algorithm which requires the approximation of the fish farm as a convex polygon.

1.3. Dissertation Outline

The outline for this dissertation is shown in Fig 1.3. Chapter 2 provides a detailed literature review on contributions of distributed formation control and path planning. The fundamental concepts leading to distributed formation control including autonomous systems, MAS, and graph theory are discussed in this chapter. A review of hybrid control architecture, existing path planning algorithms, and controllers for trajectory tracking encompass the background for path planning for the HAUCS framework.

Chapter 3 introduces the cooperative control of UAV swarms which includes the formation control framework and collision avoidance principles based on social potential functions. This chapter also discusses consensus-based control laws for pattern formation and maintenance defined by double integrated dynamics. The convergence analysis is provided

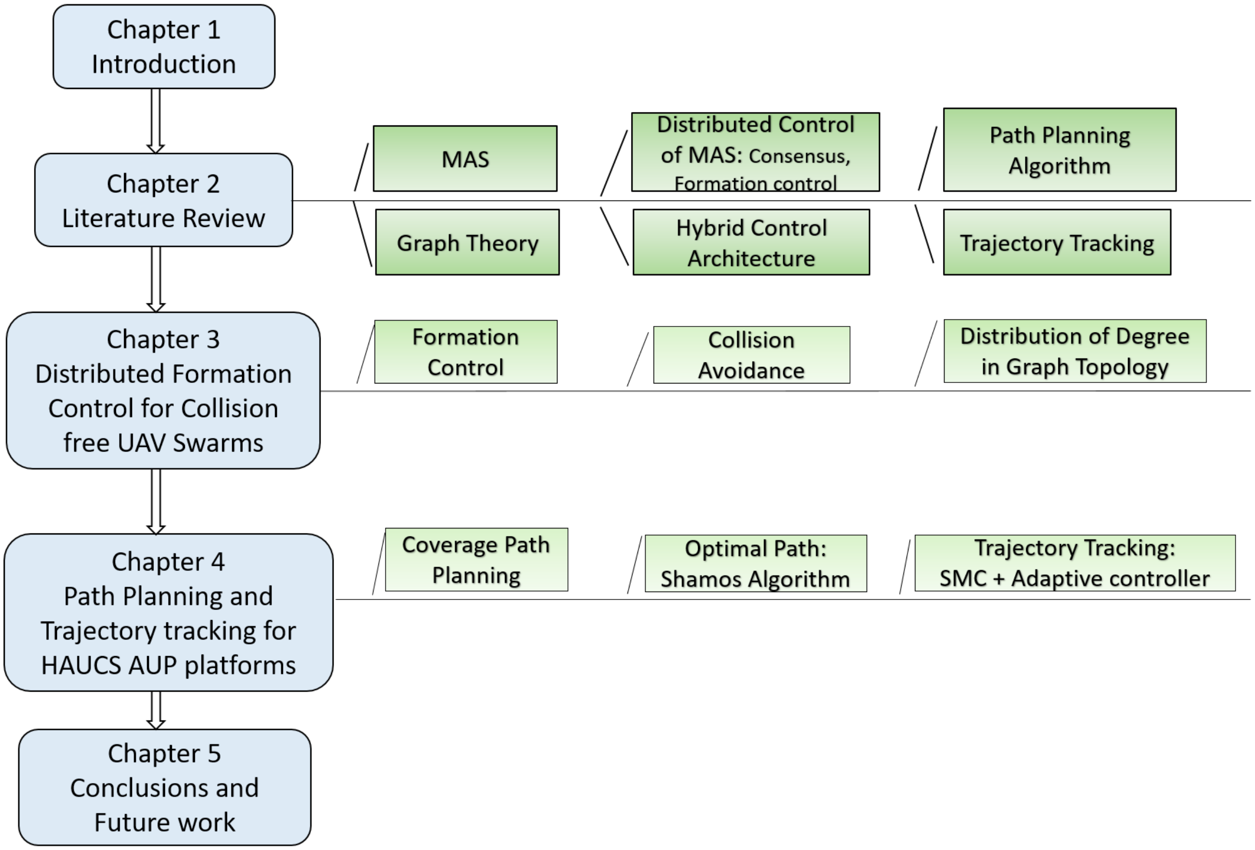

FIGURE 1.3. Dissertation Structure

along with a study of degree distribution in interaction topology. The link between the variation of degree combination of nodes in MAS and convergence is highlighted.

Chapter 4 proposes a hybrid strategy for periodic coverage of the fish farms by HAUCS AUPs. The optimal coverage method is based on the computational geometrical algorithm - Shamos Algorithm, which finds the path between sets of antipodal points on the surface of the convex polygon. The offline path generation is combined with sliding mode and adaptive controller for trajectory tracking of HAUCS platforms under uncertain wind conditions.

Chapter 5 provides the conclusions and discussions of future work for this dissertation.

CHAPTER 2

LITERATURE REVIEW

The natural phenomena of collective behavior in groups of birds, fish, and insects have served as an inspiration to the researchers to study the field of command and control of MAS. In nature, the birds in a flock or the bees in a swarm exhibit coordinated behavior that helps them in foraging for food, migration, and safety from predation. From a control point of view, such group behaviors indicate decision-making capability on both individual and collective scales. This has led to the research of control strategies for both single-entity and multiple autonomous systems.

The resemblance of autonomous systems to the natural movement of birds/bees can be attributed to the development of wireless connectivity and the use of on-board processors. This incorporation has facilitated the flow of data and complex-decision making capacities within groups of autonomous vehicles. This has opened up vast areas of possibilities in which autonomous systems are employed. They are used in places where it is difficult for humans to reach, such as inspection of faults in nuclear power plant, geographic mapping of unreachable terrains, package delivery, area monitoring in search and rescue missions, area surveillance, building inspection, precision agriculture, and many more.

2.1. Autonomous Systems

In engineering and scientific innovations throughout the world, the word "Autonomous" refers to any system that performs its tasks independent of human supervision or another engineering system. The term "Autonomous" comes from the Greek root of autos(self) and nomos(law). Autonomous systems are a class of electro-mechanical devices that consists of two components [10]:

(1) Inbuilt physical capabilities: The autonomous or unmanned system may include power manipulation or actuation devices, batteries, external sensors, and computational resources.

(2) Inbuilt control mechanisms: The intelligent control algorithms necessary for autonomous operations may consist of machine-level control, behavioral control algorithms, Artificial Intelligence (AI) algorithms, and other tracking control or mission planning algorithms.

The working of an autonomous system involving the components of physical devices and control mechanisms has been summarized as a 'concept of autonomy'. It is characterized into agent architectural model and system architectural model [11]. An autonomous agent uses at least one of the five aspects defined under the agent architectural model:

(1) Perception: The sensor data is interpreted to obtain precise information about states of other agents in the environment.
(2) Reflection: The environment model of the system is updated based on information obtained from Perception module. In case of dynamic environment, the agent needs to keep track of changing states of nearby agents. Applications of this module can been seen in collision avoidance in MAS.
(3) Goal Management: The agent prioritizes its objectives based on states of other agents and its environmental model.
(4) Planning: This model ensures that the agent executes the chosen objectives
(5) Self-Adaptation: This module monitors the states of neighboring agents and the environment to adapt in critical situations. It functions in coherence with the Goal Management and Planning functionalities of the autonomous agent.

The system architectural model explains the interaction of an agent with its environment and other components (neighboring agents and obstacles) in it. The coordination of the autonomous systems comprises the system-level architectural model.

2.2. Multi-Agent System

The interaction of multiple autonomous agents to accomplish a specific goal is a key element in the working of a Multi-Agent System (MAS). These multiple agents collaborate to perform a complex task. The interaction of autonomous agents in the MAS can be modeled as shown in figure 2.1.

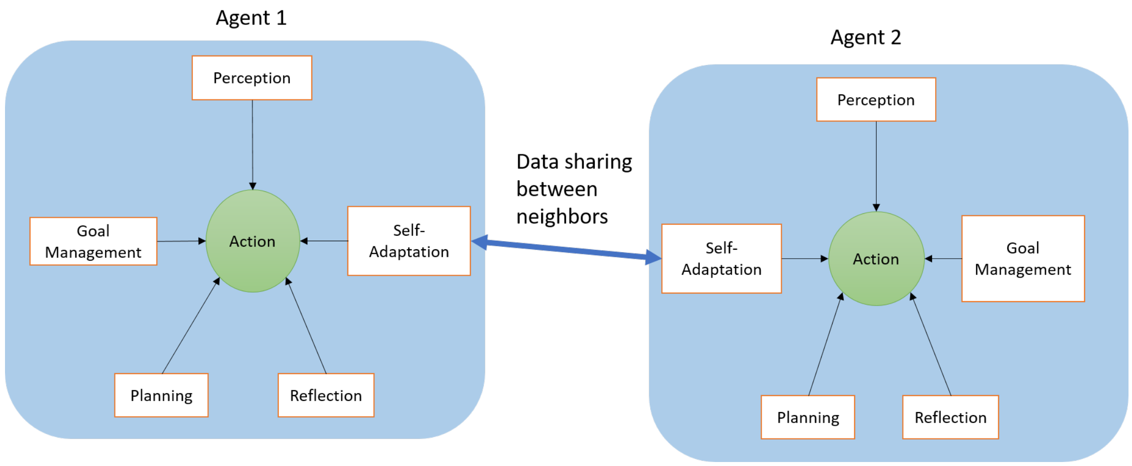

FIGURE 2.1. Structure of MAS

While an autonomous agent offers an advantage due to the feature of independent decision-making and adaptation, the performance of a collection of autonomous agents in coordination provides greater efficiency and operational capability. The features of MAS that enable to solve complex real-world problems include low cost, efficiency, reliability and flexibility. Potential applications resulting from features of MAS include area coverage and mapping of unknown terrains, transportation, defense, surveillance, and rescue missions, etc.

In a multi-agent system, a complex task is divided into simpler tasks and assigned to each agent, thus increasing efficiency. The task assignment to multiple agents also reduces the overall energy consumption as a group. Since the multi-agent system is distributive, it offers high reliability. In case one of the agents is impaired during a mission, the work can be re-distributed among the rest of the agents in MAS. This feature is important in various defense missions such as area patrolling. Other features of MAS as described in [12] include:

(1) Leadership
(2) Decision function
(3) Heterogeneity
(4) Agreement Problem
(5) Delay Consideration

(6) Topology

(7) Data Transmission Frequency

(8) Mobility

The working of a MAS highlights the collaboration among multiple autonomous agents. This has been widely explored in the collective group behavioral problems. Out of the above-mentioned features in MAS, ‘ Leadership’, ‘Agreement Problem’, and ‘Topology’ are fundamental to the applications in ‘Distributed Control of MAS’. A leader is an agent in MAS, which decides the goal and assigns the tasks to the follower agents in the group. The leader is either assigned manually by the operator or mutually decided by the agents. This is known as a leader-follower problem. In the case of leader-less applications, the follower agents autonomously take their decisions and reach the goal. For instance, the decision of an agent depends on the state of another agent in the group, when they interact to reach a consensus (or agreement). The parameters on which the agents need to agree for some applications comprise the agreement problem or consensus. The parameters can be of first-order, second-order, or higher-order. For example, in a first-order system, the agents collectively agree on the position of the vehicular formation. In a second-order system, agents reach a consensus on the position and velocity of vehicular formation. The topology in MAS represents the connectivity among the agents. It is a graph-based model that defines the network mathematically. There are two types of topology: 1) Static topology - the connectivity of the agents remains unchanged throughout the end of the mission, 2) Dynamic topology - the connectivity among the agents keeps changing. Dynamic topology is also known as ‘switching topology’.

2.2.1. MAS Applications

The applications of collective group behavior through local interaction in the MAS include combat, surveillance, space-based interferometers, reconnaissance systems, distributed sensor networks, and hazardous material handling. The key challenges in these applications lie in the development of cooperative control capabilities like formation control, rendezvous, flocking, attitude alignment, task assignment/task allocation, air traffic control, package delivery, and cooperative search [13].

A review of MAS applications that involve the collaboration among agents using consensus schemes investigated in [14] and [15] includes:

(1) Formation control: A geometric arrangement of a networked MAS is known as a formation. Unmanned vehicles, such as an unmanned aerial vehicle (UAV) or unmanned ground vehicle (UGV) connected via a wireless network are important candidates for commercial and military applications. Some of the applications such as area monitoring or 'Flying Ad-Hoc Networks (FANETs)' require networked MAS. Consensus-based distributed controllers use the relative position of neighboring vehicles to attain the formation of vehicles. Formation control was initially used in military applications during the World Wars, in Vic formation, flypast, and finger-four formation. It is a cooperative task, where UAVs interact with their neighbors to reach the formation. A distributed controller for multi-vehicle formation is provided in [16].

(2) Attitude alignment: The problem of relative maintenance of attitude between spacecraft or other vehicles is called Attitude Alignment. In [17], a formation control method is presented for attitude alignment of spacecraft using undirected graph topology. Group-level information and formation state of multiple spacecraft are also synchronized using an undirected graph in [18]. Attitude alignment is important during interferometry, where the relative attitude of spacecraft needs to be aligned during formation maneuvers.

(3) Rendezvous problem: The arrival of several agents at the same location simultaneously is called Rendezvous or 'consensus in position'. Here the interaction topology is position induced, and hence is known as an unconstrained consensus problem. A rendezvous problem for MAS is provided in [19]. Also, rendezvous or agreement over the positions of an agent in a network is provided in [20] using switching and failing communication topology.

(4) Flocking: Collective behavior of a large number of interactive agents with a common

goal is termed as ‘Flocking’. It has been inspired by the natural behavior of flocking, swarming, and schooling of birds, bees, and fishes respectively. Engineering applications based on flocking are distributed sensing in mobile networks, automated parallel delivery of payloads, military missions like reconnaissance, surveillance, and combat using cooperative UAVs. Flocking comprises a group of self-organizing mobile agents as described in [21]. The first computer animation of flocking was introduced in 1986 based on Reynolds rules that include:

(a) Cohesion: Agents stay close to their neighbors in the flock
(b) Separation: Agents avoid colliding with neighboring flock-mates
(c) Alignment: Agents attempt to match their velocity with neighboring flock-mates

2.3. Graph Theory

Graph theory plays an important role in defining the interconnection between the agents in MAS and by expressing them in a matrix form. The interaction topology of the graph can be studied for stability analysis of the formation. that deals with multiple agents and therefore multiple parameters.

An undirected graph is the most commonly used graph topology in formation control and this is attributed to the two-way information exchange between the nodes. It is given by $G = (V, E)$ where $V = 1, \ldots, N$ is the set of vertices, and $E_{i,j} = 1, \ldots, N$ is the set of edges. An edge is a link or connection between a pair of vertices or nodes and denotes the exchange of information from agent i to its neighbor j. Any graph can be mathematically defined by some matrices which are useful for complex calculations. A degree matrix denoted by $D \in R^{N \times N}$ diagonal matrix that represents the number of links connected to each node or vertex. It is given by,

$$D_{i,j} := \begin{cases} deg(v_i) \ if \quad i = j \\ 0 \quad otherwise, \quad if\ none \end{cases} \tag{1}$$

where, D is $N \times N$ matrix.

Similarly, an Adjacency matrix denoted by $A \in R^{N \times N}$ matrix that represents the information flow or connectivity between nodes (or between an agent and its neighbor). It is given by,

$$A_{i,j} := \begin{cases} 1 \ if \quad j \ is \ neighbor \ of \ i \\ 0 \quad otherwise, \quad if \ none \end{cases} \tag{2}$$

where, A is $N \times N$ matrix.

A Laplacian matrix denoted by $L \in R^{N \times N}$, captures the whole information of the graph and is essential for calculating the dynamics of the consensus algorithm [2]. It is given by, $L = D - A$ where, D and A are the degree and adjacency matrix of the same graph respectively. Also the matrix elements of L are given by,

$$L_{i,j} := \begin{cases} deg(v_i) \text{ if } i = j \\ -1 \quad if \quad i \neq j \ and \ v_i \ is \ adjacent \ to \ v_j \\ 0 \quad otherwise \end{cases} \tag{3}$$

where $deg(v_i)$ is degree of vertex i. For instance, a graph with degree, adjacency and the Laplacian matrix defined is given below for Fig 2.2

The degree matrix is written as:

$$G = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

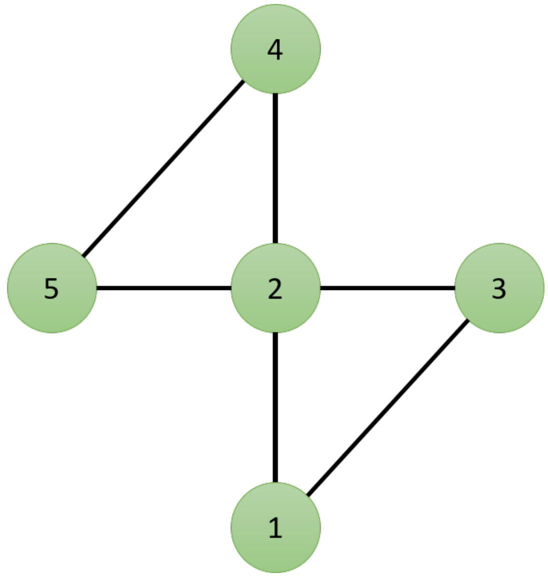

FIGURE 2.2. Example Graph Topology

Adjacency matrix as:

$$G = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

Laplacian matrix, L=D-A as

$$G = \begin{bmatrix} 2 & -1 & -1 & 0 & 0 \\ -1 & 4 & -1 & -1 & -1 \\ -1 & -1 & 2 & 0 & 0 \\ 0 & -1 & 0 & 2 & -1 \\ 0 & -1 & 0 & -1 & 2 \end{bmatrix}$$

For a given graph topology and its Laplacian matrix, L:

(1) L is a symmetric matrix

(2) The eigenvalues of L matrix provide a great deal of information about the network. For example, the second smallest eigenvalue (Fiedler value) represents the algebraic connectivity of a network.

(3) L is a singular matrix, i.e. the determinant of L is zero.

(4) The diagonal elements of L are positive and the off-diagonal elements are negative.

2.4. Formation Control of MAS

Formation control is an important application of MAS where a group of autonomous agents follows a predefined trajectory while maintaining a desired geometrical pattern. The advantages of moving in formation include reduced system cost, increased robustness and efficiency, reconfiguration ability and structural flexibility of the system [22]. The two broad categories defined in formation control include:

(1) Formation producing: This encompasses a formation control design that enables the agents to reach formation without any group reference.

(2) Formation tracking: This includes algorithm design that requires agents to reach formation by following or by keeping track of a group reference.

Formation tracking is considered more challenging than the formation-producing problem because of the group reference used. Also, the design algorithms for the former cannot be applied to the latter. By definition, formation is the attainment of a desired geometric order by agents. The resemblance of the agents with nodes of a graph prompted researchers to use graph theory as a tool for analyzing the formation problem.

2.4.1. Classification based on Controllability Analysis

In formation producing, the principles of graph theory are frequently used for the stability and controllability analysis of formation. The degree matrix, adjacency matrix, and Laplacian matrix serve as tools for the same as stated in [23]. The eigenvalues of the L matrix reveal much information about the stability of the network. A fixed network topology displays:

(1) The existence of at least one zero eigenvalue.

(2) The presence of at least one pair of eigenvalue on the imaginary axis in the system matrix of a linear closed loop system.

However, the complex analysis and working of switching network topology makes the applicability of these properties very difficult and thus viewed as an open research problem [24]. Research on formation stabilization where the topology is represented by an undirected graph has been discussed in [22]. It states that spectral analysis of a graph plays a vital role in the control of multi-agent formation. The smallest positive eigenvalue of the Laplacian matrix decides the time taken to reach formation by the agents. It is also required that the graph is well connected for the usage of a linear stabilizing feedback law.

Lyapunov function approach is the most favored method used for stability analysis of complex dynamical systems and control theory because the analysis of nonlinear systems is done easily with the Lyapunov function than the graph theory approach (or matrix theory approach). The types of formation producing that have been studied with this approach are the inverse agreement problem, the leaderless flocking and stabilization, and the circular formation problem. A brief review of these problems can be found in [24].

The formation tracking problem has also been studied by the graph theory or matrix theory approach. The design of a control system that allows the agent to keep track of the reference to reach the desired position is interesting and has been discussed in detail in [24]. The difference between the state of the agent and the reference is taken as an error. The goal of the system would be to minimize the error and reduce it to zero. However, this method can only be applied to the formation tracking system. This makes it easy for the formation tracking system to be solved under a switching network topology.

Lyapunov Function is also widely used for the stability analysis of systems in the domain of formation tracking. An example of flocking with dynamic group reference has been discussed in [24] where the agents need to move cohesively along the group reference. This study of a system with dynamic group reference is more challenging than an unchanging group reference. This makes a leader-follower problem more complicated than leaderless flocking. The Lyapunov function has also been applied to systems with variable structure-based control laws to get better results.

2.4.2. Classification Based on Control Strategy

(1) Behavior-Based or Potential Based: Behavior-based control strategy is used in MAS to fulfill navigational goals such as obstacle avoidance, collision avoidance, and maintaining the formation, as well. It is always combined with the potential field approach. This control strategy enables individual agents or robotic vehicles to concentrate on the inputs received by their sensors and act on them. Thus, all the agents in the formation respond to information obtained from their surrounding areas and ensure full coverage of the formation. This kind of control action can be observed in air force fighter pilots who restrict their visual and radar range to an area of terrain based on their current positions. Applications of these formation methods can be seen in search and rescue operations and security patrol as mentioned in [25].

(2) Leader-Follower: The formation of MAS using the Leader-follower control method has at least one leader with the rest of the agents as followers. The control design is such that followers track the position of the leader and the leader tracks its prescribed trajectory. This method is an example of formation tracking with a reference. In [22], formation control with two types of feedback controllers is discussed. $l - \psi$ controller: The desired length of l_{12}^d and a desired relative angle of $\psi_1 2^d$ is maintained between the leader and the follower. Two-wheeled ground mobile robots are examples where input/output feedback-linearization can be used to design a controller where l_{12}^d and ψ_{12}^d can achieve convergence.

$l - l$ Controller: In the example mentioned in [22], the formation contains two leaders and one follower. The follower robot is controlled to track and follow the leaders. The desired length l_{13}^d and l_{23}^d between the follower and leaders is maintained. The input/output feedback-linearization can be used in $l - l$ controllers as well. Mostly, nonlinear systems can be controlled by the feedback linearization method. The conversion of a nonlinear system into a linear system is done by changing some of the variables and input conditions. Various tools and theories of linear control methods can be applied to develop a stabilized system. It can be applied to both single-input single-output (SISO) systems and multiple input multiple output systems (MIMO).

(3) Generalized Coordinates: The generalized coordinates-based control strategy uses the vehicle's or agent's location (L), orientation (O), and shape (S) with respect to the reference point set in the formation. The L, O, and S coordinates are used to describe the agent's trajectories as mentioned in [22].

(4) Virtual Structure: Formation control by the virtual structure method was introduced in [26]. This method is used in applications where a fixed formation geometry is required. Spacecraft application in deep space is an example. Also, in laser interferometry, the instruments are required to fixed kilometers apart in space to get a proper reading. The idea for this concept was derived from the behavior of a rigid body. Particles in a rigid body are in a fixed geometry and any force or disturbance made to one particle will propagate to all other particles comprising the body. Any robotic system build using this concept was thought to be highly desirable as discussed in [26]. The controller of the virtual structure method follows three steps. Firstly, the desired dynamics of the robotic structure to be built is defined. Secondly, each robot or agent is made to follow the desired motion of the whole virtual structure. Thirdly, controllers to track each agent are designed. Further details are discussed in [22].

(5) Model Predictive Control: One of the recent coordinated control laws is the model predictive control (MPC). Agents or robots are controlled locally by defining a local control law. The presence of inter-vehicle communication and the distributed nature of the control design takes care of the total formation. This is desirable because a single agent cannot have access to a large-scale formation of agents. This kind of control mechanism has been used in [9] on simple 1D vehicles.

2.4.3. Classification Based on Sensed and Controlled Variables

Any formation control scheme consists of variables that are sensed by agents and the variables that are actively controlled. This is defined in terms of sensing capability and interaction topology of the agents. The sensing capability of the formation is dependent on the types of variables that are sensed. Also, the topology formed by the agents describes the type of controlled variables needed. A detailed review is presented in [27]. There are ways in which the sensed and the controlled variables can be alternatively used to decide the different types of controllers. For instance, when the distances between the agents are controlled, then the agents need to communicate with each other. Thus, the system would act as a rigid body. On the other hand, when the positions of the agents are controlled directly, then the agents need not communicate with each other. These kinds of variations are used to decide on the classifications such as:

(1) Position-Based Control: A formation control scheme that has position-based control causes its agents to actively control their positions with respect to the global coordinate system. The two components of this method are sensing capability and interaction topology. Interaction among agents and feedback taken from agents by a global coordinator are the two important steps in position-based formation control.
(2) Distance-Based Control: Distances between agents are actively controlled to reach the formation. Each agent figures out relative positions with respect to their neighbors based on the local coordinate system. Sensing capability requires that agents know the local coordinate system. Since the desired distances between agents (or desired formation) are fixed, the interaction topology should be rigid. The agents use translational and rotational motions to attain formation.
(3) Displacement-Based Control: The displacements between agents and their neighbors are actively controlled to achieve the desired formation. The input is in the form of the desired displacement based on the global coordinate system. The agents need to know the orientation of the global coordinate system. The choice of a control method is based on many different criteria that can be varied as per application. Distance-based and displacement-based control using single and double integrated dynamics have been given in detail in [27]. A review of the analysis comparing sensing capability and interaction topology has also been mentioned.

2.5. Hybrid Control of MAS

The Internet of Things (IoT) is a standard that describes the connectivity between 'things' or 'objects' around us. It incorporates the usage of wireless communication to connect a variety of these things around us - such as sensors, actuators, mobile phones, autonomous vehicles, computers, radio-frequency identification tags (RFID), etc. These "things" interact with each other to accomplish a common objective [28]. The architecture of IoT maps to an environment that is decentralized and distributed to have greater flexibility, agility, and dependability. This makes multi-agent systems (MAS) a suitable candidate for IoT applications. The usage of MAS increases the autonomy and flexibility of the IoT application. Currently, most of the environments in which we live such as at home, at work, in parks, at hospitals, etc have a basic level of intelligence. This means that the environment does not offer any communication capabilities that connect the 'things'. The usage of IoT would make smart environments where the 'things' can communicate with each other and therefore improve the quality of our lives.

The possible applications of IoT can be found in the following areas:

(1) Transportation and Logistics
(2) Healthcare
(3) Smart Environment (Office, Home)
(4) Social Network

Among the potential applications of IoT, the transportation and logistics domain offers enormous advantages in the field of "Precision Agriculture" (PA).

Precision agriculture (PA) is the application of robotic field machines and information technology in agriculture [1]. The application of robotic technologies in agriculture as described in [29] improves the productivity and efficiency of these farm's manifolds. One of the fast-growing sectors of agriculture mentioned in [30] is aquaculture farming. Management of water quality is an important aspect of aquaculture fish farming. The amount of dissolved oxygen (DO) in water is crucial for maintaining the quality of the fish stock. The current method of water quality management involves human operators who drive a sensor-equipped truck along individual ponds to measure DO levels. The periodic and continuous monitoring of very large farms (greater than 400 hectares) require multiple human operators to use several sampling instruments. This would increase the associated labor and equipment cost, thereby hampering the monitoring process.

Hybrid Aerial/Underwater RobotiC System (HAUCS) is an application of IoT framework that allows for collaboration between robotic systems, central control station, weather station, machines, and human operators. The deployment of autonomous unmanned platforms (AUP) such as HAUCS could drastically improve the current labor-intensive and resource-constraint operations. HAUCS AUPs consist of unmanned robotic vehicles along with submerged underwater sensors, that fly in the air and travel on the water surface to monitor critical parameters.

2.5.1. Hybrid Control Architecture

The design of control architecture is a challenging task for the performance of an IoT framework. The present control systems have advantages and disadvantages that affect the functional capabilities of a robotic system. A hybrid control structure is a combination of deliberate, reactive, distributed, and centralized approaches in the robotics paradigm. The hybrid model utilizes a central module for supervision, generation, and execution of commands, and distributed module to process the sensory inputs from field robots. This provides a goal-oriented, real-time responsive, and tele-operable control system architecture as proposed in [31].

The robotic paradigm is described as a relationship between the basic elements of robots, which are sensing, planning and acting [32]. These elements are integrated into a robotic system to process the data and make decisions. The control architectures used for the operation of robotic systems are as follows:

(1) Deliberative: In the deliberative or hierarchical system the individual processes are run sequentially to get action. Initially, all the sensing data is gathered by the sense module. The control command for the path to be traversed from initial state to goal state is generated by the plan module. This path is run by the execution or

act module. This method assumes a global whole world approach. The drawback of the method lies in the slowness of the combinatorial explosion of the optimal path search process every step of the way. In a robotic system designed using this approach, the robot would stop and deliberate its next step before taking action as discussed in [31]. The deliberative process is shown in figure 2.3

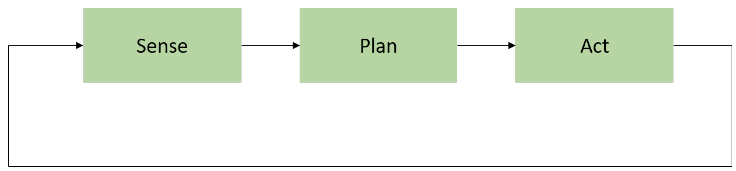

FIGURE 2.3. Deliberative Paradigm

(2) Reactive: A reactive system responds directly to the existing stimuli in the environment. This method acts directly on the current sensory input and therefore lacks a global viewpoint required to determine possible future states. The concept of a reactive system is shown in the figure. 2.4.

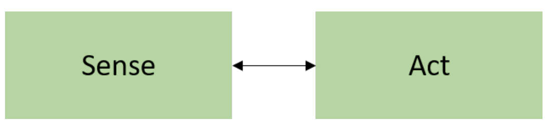

FIGURE 2.4. Reactive Paradigm

(3) Hybrid: A system that is both goal-oriented and reactive. It is needed for the effective performance of a robotic process. Such a system would also include features of centralized, distributed, top-down, bottom-up, command arbitration, behavioral selection, and sensor fusion models. A deliberative (planning) top layer would generate the high-level navigation path whereas the low-level reactive layer would provide reactive responses. The high-level navigation path is presented as simple executable tasks that direct the robot to the goal. The low-level reactive module receives the information of the path and generates required control actions for execution. This presents a teach mode of operation (by high-level module) and playback mode of operation (by low-level module) that is required in the case of tele-operation. The

hybrid paradigm also enables self-supervised goal-oriented autonomous operations to handle various uncertainties. The hybrid paradigm is shown in the figure. 2.5.

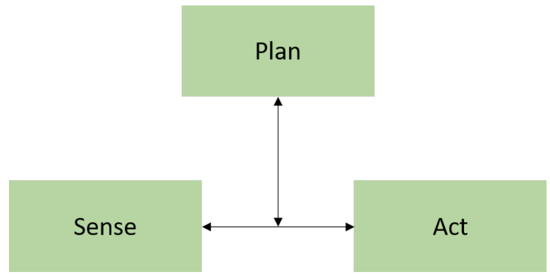

Figure 2.5. Hybrid Paradigm

2.5.2. Path Planning of MAS

The usage of multiple autonomous agents such as HAUCS requires algorithms and control strategies that decide the path of the agent. Mostly, the agents are desired to move from an initial configuration to a goal configuration. Different algorithmic procedures have been proposed by researchers [33], for path planning in environments with/without obstacles. Generally, the path planning methods provide the path for the individual AUP to complete the mission under certain environmental conditions. The environment can be categorized as static or dynamic. Path planning in a static environment assumes no moving obstacles, and the trajectory of the agent to its destination is predetermined. In a dynamic environment, which may include moving obstacles or neighboring AUPs/agents, there is a need for dynamic path planning techniques [33].

2.5.3. Path Planning Algorithms

Some of the path planning algorithms are Traveling Salesman Problem (TSP), Geometrical Algorithms, Artificial Potential Functions, Coverage Path Planning (CPP), Distributed Dynamic Programming, and Resource Allocation protocols.

The Traveling Salesman Problem (TSP) is a task allocation algorithm that generates an optimal path for a vehicle to travel across some target locations with a minimum cost. It is an NP-hard problem in combinatorial optimization. If the AUPs have limited energy and limited weight carrying capacity, characterized as a Vehicle Routing Problem (VRP).

Present applications of TSP/VRP algorithms are in package delivery and data collection [34].

Collision avoidance is critical in any path planning procedure. Artificial Potential Functions (APF) are a set of algorithms frequently used to navigate autonomous agents [35, 36]. This method uses the concept of potential functions from magnetic and electric fields. Here, each vehicle or agent is taken as a point charge in space that gets attracted or repelled from nearby agents based on their distance. This results in a push/pull dynamic between agents. The attractive or repulsive force, typically defined as the negative gradient of a suitable potential function, is applied to each agent. Applications of APF can be found in path planning [35] and task allocation [36]. APF algorithms have high scalability and low bandwidth use [37].

Coverage Path Planning (CPP) algorithms determine the optimal route of an AUP to cover a desired area or space while avoiding obstacles. The algorithms also consider environmental conditions such as weather changes. The agents are instructed to make continuous and sequential movements without overlapping their paths. CPP is generally used in mapping, crop monitoring, and land assessment [34]. Approaches to CPP are categorized as randomized and complete. A randomized approach does not consider the geographical information of the farm and, therefore, takes a long time to cover the whole area. On the contrary, in a complete approach, the coverage area is decomposed into cellular regions, and an optimal path is traversed to cover all the cells (i.e., ponds). Details about cellular decomposition and optimal path searching can be found in [38, 39]. Firstly, the exact cellular decomposition technique categorizes the desired area into non-overlapping regions or cells. Here, the adjacent cellular decomposition is represented by an adjacency graph In the next step, the planner computes a sequence of cells based on an exhaustive search on the adjacency graph. The cells are covered by robotic vehicles in a back and forth or zig-zag pattern. The two methods for off-line cellular decomposition include: 1) trapezoidal decomposition - In this method, each cell is in the shape of a trapezoid and guarantees complete coverage based on exhaustive search. However, the coverage path calculated is longer since, the trapezoidal method only generates large number of convex cells. and 2) boustrophedon decomposition - The drawback of trapezoidal method is overcome be this method that reduces the number of generated cells. The term boustrophedon comes from a Greek word meaning, “the way of ox”, representing the back and forth movement of ploughing by an ox. This is an improved version of trapezoidal method, where, only those vertices are considered that allow a vertical line to pass in both upward and downward direction. These vertices are called critical points. This improvement reduces the number of generated cells for area coverage.

Geofencing is a recent method of path planning that has found applications in UAS traffic management. In geofencing, airspace is divided into several zones used by an individual AUP or team of AUPs. This system is designed for traffic management across the growing number of AUPs in the airspace. Some scenarios of interacting geofences for multi-stage flight plans have been detailed in [40]. A geofencing-based path planner is presented in [41]. Bio-inspired algorithms like Ant Colony Optimization are proposed in [42] for farmland monitoring. A geofence can be defined as a Static geofence and Dynamic geofence. A static geofence is an area with fixed boundaries which is always active. It can represent an airspace in an airport, a military base, or a stadium. A dynamic geofence is one which is dynamic with changing home locations, but not necessarily always active. It is further classified as durational and trajectile geofences.

2.5.4. Shamos Algorithm

Computational geometry is the study of algorithms representing the concepts of geometry in Computer Science. Theoretical analysis of algorithms that are based on concepts of computational geometry is called “Algorithmic Motion Planning”. This includes motion planning of robots/agents based on methods in combinatorial and computational geometry related to the arrangement of curves and surfaces. The term “Computational Geometry” was first coined in the Thesis of Michael Shamos in 1978. It contained a detailed analysis of algorithms representing geometrical concepts of points, surfaces, convex polygons, and convex hulls. Shamos Algorithm proposed the concept of “Antipodal Points”. The points on the surface of a convex polygon that admit two parallel lines of support passing through them, such that the whole polygon lies between them, is called antipodal pair of points. The diagram representing antipodal points on a polygon is shown in Fig 2.6. The Shamos Algorithm 1 determines the all pairs of antipodal points for a given convex polygon.

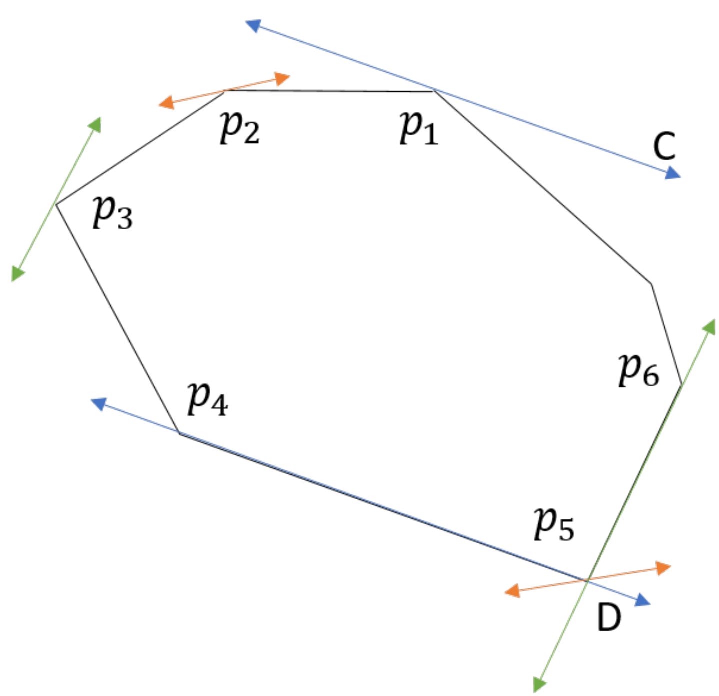

FIGURE 2.6. Antipodal Points

Algorithm 1: Shamos Algorithm

Data: V

Result: A

`/* Find initial antipodal pair by locating vertex opposite` p_1 `*/`

i $\leftarrow$ 1 ; j $\leftarrow$ 2 **while** *(angle(i,j) ¡* π **do**

  j $\leftarrow j+1$ current $\leftarrow i$ A $\leftarrow (i,j)$

end

`/* Loop on j until whole polygon is scanned */`

while $j \neq n$ **do**

  if *(angle(current,i+1)* $<$ *angle(current,j+1))* **then**

    j $\leftarrow j+1$

    current $\leftarrow j$

  else

    i $\leftarrow i+1$

    current $\leftarrow i$

  end

  A $\leftarrow (i,j)$

end

`/* On parallel edges */`

if *(angle(current,i+1)* $=$ *angle(current,j+1))* **then**

  A $\leftarrow (i+1,j)$; A $\leftarrow (i,j+1)$

  A $\leftarrow (i+1,j+1)$

end

if $current = i$ **then**

  $j \leftarrow j+1$

else

  $i \leftarrow i+1$

end

The polygon vertices (p_1, p_4) are an antipodal pair. Similarly, (p_1, p_5), (p_3, p_6), (p_3, p_5), and (p_2, p_5) are other pair of antipodal points. The lines C and D passing through vertex $p1$ and edge (p_4, p_5) are parallel lines of support.

The convex polygon is defined as $Q = (V, E)$, where $V = 1, 2, ..., n$ is the set of vertices and $E = (1, 2), (2, 3), ..., (n, 1)$ is the set of edges of the polygon in clockwise direction. The rotating lines of support are metaphorically viewed as calipers used to measure the angles, hence it was also named as "Rotating Calipers Algorithm". The rotation of the calipers are shown in Fig 2.7

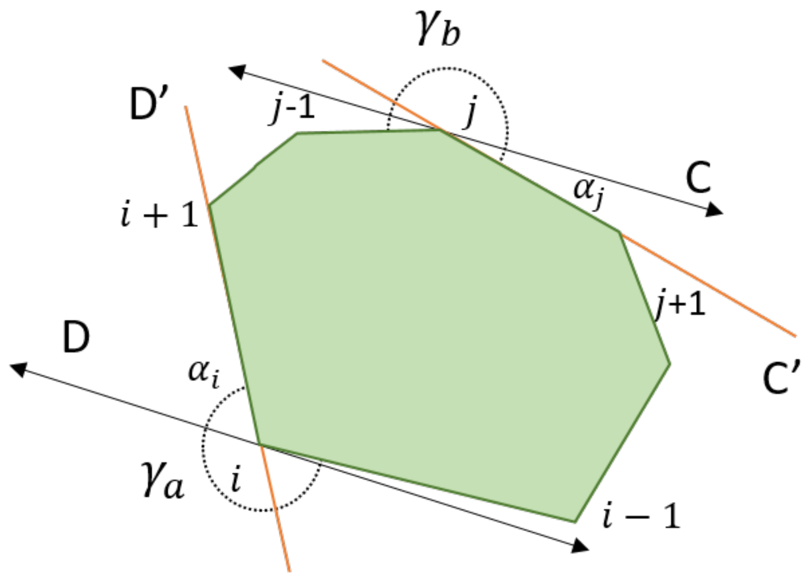

FIGURE 2.7. Rotating Caliper

The antipodal points are computed by rotating the parallel lines of support in a counterclockwise direction along the surface of the polygon until an edge is found. In Figure 2.6, (i, j) are an antipodal pair that admit lines of support, C and D. When these lines are rotated counterclockwise, line D touches vertex $i-1$ sooner than line C touches vertex $j-1$. Therefore, when the lines are still parallel during rotation, vertices $(i-1, j)$ form the second pair of antipodal points. The computational complexity of the Shamos Algorithm is $O(n)$, as there are n vertices in the polygon. Shamos algorithm is used to find the best path for every antipodal pair, thereby giving the optimal coverage path for a convex polygon.

2.5.5. Trajectory Tracking

The tracking of the path generated by a path-planner is a crucial aspect of automated control of vehicles. Trajectory tracking is a challenging task because of non-linear dynamics, sensor disturbances, noise, unknown parameters and safety criticality. Trajectory planning and trajectory tracking using feedback controllers are the two aspects of controlling an autonomous system as described in [43].

Some of the control methods useful in trajectory tracking include:

(1) Sliding Mode Control (SMC): It is a nonlinear control technique that drives the system states onto a particular sliding surface. The controller applies a discontinuous control input to the system such that it slides along the prescribed path. When the system reaches the prescribed path (or sliding surface), the controller keeps the states close to the neighborhood of the sliding surface. The two parts of the SMC include - 1) Selection of the sliding surface and 2) Selection of the control law that makes the switching surface attractive to the system state [44]. SMC provides the advantage of not linearizing complex dynamics and has good tracking as mentioned in [45]. An SMC is also insensitive to various uncertainties. The control input for the first-order system is a discontinuous signum function, given by, $u = -USgn(\sigma)$. The Sliding mode control is shown in Fig 2.8.

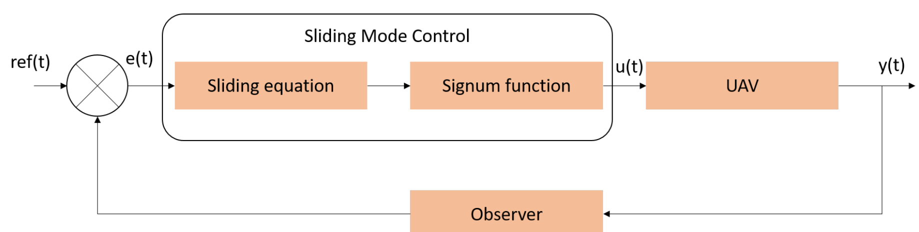

FIGURE 2.8. Sliding Mode Control

(2) Adaptive Control: An adaptive controller can modify its behavior in response to change in uncertain or time-varying parameters in the system. It is a nonlinear feedback control that operates in two stages. In the first stage, the slow-changing states are viewed as parameters with a fast time scale. The second stage involves a slower time scale for updating regulator parameters. The goal of the adaptive controller is to compensate for uncertain parameter variations due to disturbances acting on the process. The adaptive controller is shown in Fig 2.9.

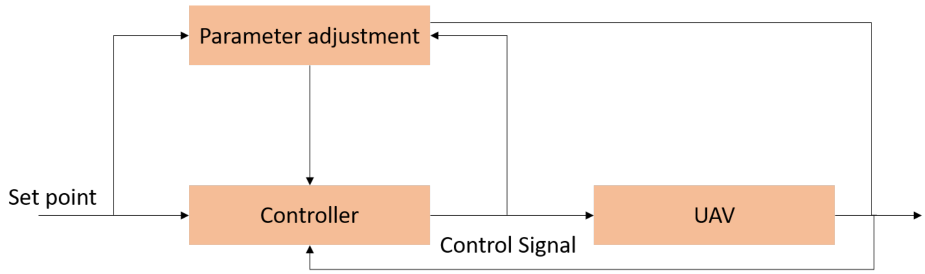

FIGURE 2.9. Adaptive Control

(3) Backstepping Control: This technique was built in 1990 by Peter V. Kokotovic for stabilizing special classes of non-linear dynamical systems. A backstepping controller follows a recursive design procedure to stabilize sub-systems and 'back out' to progressively stabilize each of the outer sub-systems. The process is backstepping ends when the final outer sub-system is reached. Here, the design of a control Lyapunov function is linked to the feedback controller to ensure the global asymptotic stability of the strict feedback systems. The adaptive and robust backstepping control methods are effective in dealing with uncertainties in the system [46].

CHAPTER 3

DISTRIBUTED FORMATION CONTROL FOR COLLISION FREE UAV SWARMS[1]

3.1. Introduction

Cooperative control in the field of multi-agent systems (MAS) is inspired by natural behavior such as flocking exhibited by birds. It describes a coordinated movement of individual agents of a group through shared interactions. Such behavior achieved through local interactions offer advantages such as flexibility, adaptability, robustness, coverage, low energy, and high performance. The effectiveness of such collective behavior lies in the dynamics and information exchange among the interacting agents. Cooperative control strategies benefit many engineered systems including sensor networks, robots, and autonomous systems with applications to intelligent transportation, defense, security, and surveillance.

3.1.1. Consensus-Building

Cooperation among the agents of a MAS leads to consensus for a desired state. Consensus-building is a process to reach an agreement among agents through information sharing. Consensus-building has its roots in control theory [47], [48] and found applications in many other fields including sociology, parallel computing, power systems, biology, sensor networks, and robotics [49].

3.1.2. Formation Control Strategies

One of the classical problems in cooperative control is formation control. In this problem, consensus-building leads to the desired shape and alignment of the formation. In general, formation control strategies are distributed and comply with the chosen interaction topology. They can be broadly grouped into two classes: formation producing [27] and formation tracking [24]. The former produces a formation in the absence of a group reference while the latter follows a desired group reference. Formation control involves each vehicle

[1]This entire chapter is reproduced from S. Mukherjee and K. Namuduri, "Joint Flocking and Deconfliction in Unmanned Aerial Vehicle Swarms," MILCOM 2019 - 2019 IEEE Military Communications Conference (MILCOM), 2019, pp. 49-55, doi: 10.1109/MILCOM47813.2019.9020935, with permission from IEEE.

maintaining its relative position within the geometric pattern while flying together. The three main approaches to formation control are as follows: [50, 22]:

(1) Behavioral approach: The vehicles are expected to follow a set of desired behaviors on a priority basis. Such behaviors include collision avoidance, obstacle avoidance, and tracking among others. [51, 52, 53].

(2) Virtual approach: The group of vehicles is controlled to behave as a single rigid formation. Each vehicle can be thought of as a particle of an object that constitutes a formation [54, 55].

(3) Leader-Follower approach: One vehicle is designated as the leader and others as followers. The leader creates its own desired track independently and the followers create their tracks maintaining their relative positions with respect to their neighbors within the flying pattern [47, 48, 49, 27, 56, 57, 58, 59, 60].

3.1.3. Leader-Follower Model

This paper is focused on the Leader-Follower (L-F) strategy. Strategies for a L-F formation control include usage of artificial potentials fields, adaptive vision-based control, and sliding mode controller as discussed in [50] , [61], and [62] respectively. When networked agents as in a MAS are considered, consensus-based control laws are more suitable for L-F structure implementation. The concept of multi-agent coordination in terms of consensus is presented in [63].

Formation control may or may not include an explicit formulation for collision avoidance. Typically, it is implemented between vehicles as a pair-wise strategy. But, a distributed approach to collision avoidance is needed for MAS. Social potential functions that involve attractive and repulsive forces among agents provide a means to design a distributed solution to formation control with collision avoidance.

3.1.4. Graph Representation of UAV Swarms

A UAV swarm is a MAS consisting of communicating agents with network connectivity that can be described by a graph topology. Graphical representation emphasizes the direction of information flow between agents that could be directed or undirected. Another way to classify the topology is by considering a change in pairs of interacting agents over the duration of formation control. In fixed topology, each agent has the same set of neighbors, while in switching topology the agents dynamically change their neighbors. A novel control protocol using directed and switching graphs proposed in [64] solves an L-F based formation control in three dimensions. An interesting work by Wei Ren [65] analyzes the consensus protocol with bounded control inputs and without relative velocity measurements. It is applied on a system of double integrated dynamics with an undirected graph and is further extended to a directed graph with group reference velocity.

3.1.5. Dynamics and Connectivity in MAS

The two important components for consensus problem in MAS are dynamics and interconnections. Agents can have first-order, second-order or higher-order dynamics. They can also be classified as linear or non-linear, continuous or discrete models. These isolated agents are interconnected by an underlying structure of their communication links. This structure can be represented in the form of a graph. Algebraic graph theory is a useful tool in designing cooperative control strategies for dynamical systems represented by MAS. In this representation, each agent is considered as a node or vertex in the graph that represents a MAS. The agents are connected to one another through communication links represented as edges in the graph. Application of graph theory in MAS control is reviewed in [66]. Matrices such as the adjacency and Laplacian can be used to describe the connectivity and to analyze stability and convergence aspects of a MAS.

3.1.6. Stability and Convergence

A significant amount of work has been done towards stability of formation control protocols and maintaining a formation pattern. However, in practical scenarios, other parameters involving communication links, play a major role in estimating the convergence rate of MAS. One of the potential directions of research involving interconnections of graph is the optimization of parameters in formation control[63]. In this work, we observe the effects of varying interconnection patterns on the convergence of MAS.

3.1.7. Contributions

This paper first presents a distributed model for MAS characterized by double integrated dynamics and provides a solution to formation control by combining consensus laws with social potential functions. This is an extension to our previous work [67]. Double integrated dynamics is a second order MAS that takes into account the position and velocity of an agent for generating the control inputs. The model makes use of the fundamental concepts including consensus strategies [68], [23], leader-follower structure [69], social potential functions, and algebraic graph theory to derive the necessary control laws to jointly address flocking and deconfliction in the formation control problem.

Second, it develops concepts of collision and interaction zones to classify the neighboring area of each agent. This classification provides a means to identify those vehicles that may potentially collide or those vehicles that may be on the verge of going out of communication range of an agent. This classification enables us to apply attractive and repulsive forces appropriately. It also introduces the definitions of logical and physical neighbors of an agent in the context of collision avoidance and connectivity.

Third, it investigates the effects of graph topology on the formation control process motivated by importance of equal convergence in MAS. Through experimental results it has been demonstrated that the degree distribution in the graph representing MAS, affects the rate at which formation is attained.

3.1.8. Organization

The organization of this paper is as follows: In section II, the background is presented. The model for formation control is provided is section III. Section IV includes convergence analysis for the proposed model. The numerical analysis and results are discussed in section V. Finally, the conclusion and future works are provided in section VI. The terms UAV, vehicle, and agent will be used interchangeably throughout the paper. A fixed and undirected graph topology is considered to describe the geometric formation.

3.2. Background

This section outlines the related literature on the fundamental concepts that were introduced in the previous section.

3.2.1. Graph-Theoretical Concepts

A MAS can conveniently be represented as a graph which captures the information flow among individual agents. The graph, $(G = (V, E))$, consists of a set of vertices (V) representing all agent in the MAS. The vertices $(V = (v1, v2, ..., v_N))$ are linked to their neighbors through edges $(E = (e1, e2, ..., e_N))$. A graph is represented as a matrix for MAS analysis purposes. The $N \times N$ adjacency matrix A indicates the connectivity between nodes as:

$$a_{i,j} = \begin{cases} 1, & (i,j) \in E, \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

The degree matrix D is a $N \times N$ diagonal matrix denoting the number of neighbors of each node.

$$deg_{i,j} = \begin{cases} 1, & i = j \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

The Laplacian matrix L is computed as $D - A$ with individual elements $(L_{i,j})$ defined as follows:

$$L_{i,j} = \begin{cases} deg_{i,j}\,, & i = j \\ -1, & i \neq j \text{ and } v_i \text{ is adjacent to } v_j \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

3.2.2. External Factors: Noise and Communication Failures

In the real world, a continuous process like formation control is affected by external factors such as noise, network link failures, etc., which lead us to the problems of collision and connectivity failure among agents. Existing solutions highlight the importance of avoiding an inter-agent collision while maintaining network connectivity. Separation distance among agents characterizes both collision and connectivity zones. Social potential field methods have been widely considered [70, 71, 72, 73] for maintaining the separation distance among agents.

3.3. Problem Definition

This section proposes a cooperative control model with collision avoidance for an UAV swarm. The objective is to generate and maintain a desired formation pattern while ensuring inter-agent collision avoidance. As shown in Fig. 3.1, the overall control scheme applied to each agent includes four components - Communication Network (CN), Formation Generation Module (FGM), Collision Avoidance Module (CAM) and, the Actuator. Each module implements the desired swarm behavior. The design incorporates geometric aspects of network along with vehicle level control. The CN module contains the graph topology of connected agents and their information states. The FGM implements the consensus-based formation control laws for individual agents. The CAM calculates the repulsive force applied on the agent to avoid possible collisions. The Actuator implements the displacement-based control to achieve desired formation through sensing and interaction [27].

3.3.1. Communication Module

The communication network module contains information about the graph topology connecting the agents. It is assumed that the relative position, $p_i(t) - p_j(t)$ and the relative velocity, $\dot{p}_i(t) - \dot{p}_j(t)$ information is available to each agent i. This information serves as input to the FGM and CAM. The control input (u_i) is applied to the individual agents. The position and velocity output is fed back to the CN module.

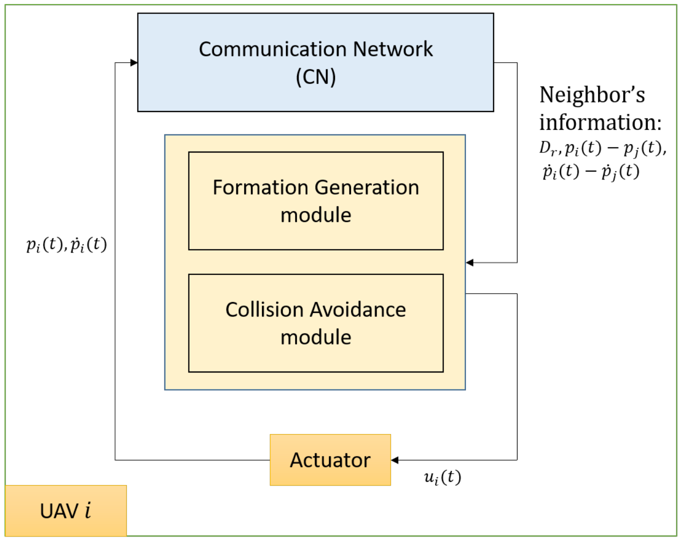

FIGURE 3.1. Control Scheme

(1) Network Topology The underlying graph framework connecting the agents is assumed to be undirected and time-invariant. It facilitates bidirectional information exchange between agents and their neighbors. An example of a MAS with five nodes is illustrated in Fig. 3.2.

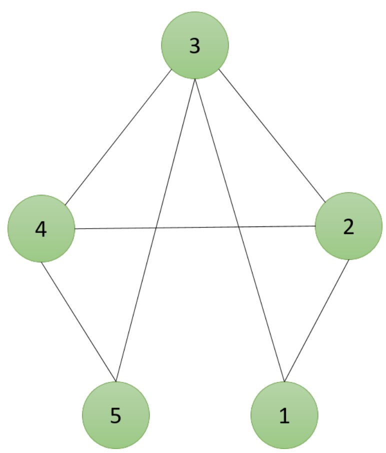

FIGURE 3.2. Graphical representation of a network

(2) Neighborhood Classification In a MAS, communication links connecting the individual agents determine the logical neighbor set, N_i of each agent i. The agents are assumed to be in their respective range of communication throughout the formation

process. Each agent i would exchange information such as position and velocity with its neighboring agents. However, during the course of formation, other agents which may not be part of the N_i set may come within the collision range of agent i. Since, the physical neighborhood of each agent i is time-varying, it is important to identify the agents which are in the collision range. To address this issue, we introduce a zone classification model comprising of collision and interaction zones as shown in Fig. 3.3. This zone classification allows us to apply formation control laws and repulsive forces selectively.

The area around every agent is separated into two zones. An agent is denoted as a point in the center, surrounded by solid red and dotted red concentric circles of radii r and r' respectively. These two red circles together form the collision zone of an agent. The repulsive force applied on an agent will depend on the presence of neighbors in the inner or outer collision zones. The interaction zone is shown by the solid blue circle of radius R. Neighboring agents falling in this zone are only subject to consensus based formation control laws. The two zones are defined as follows:

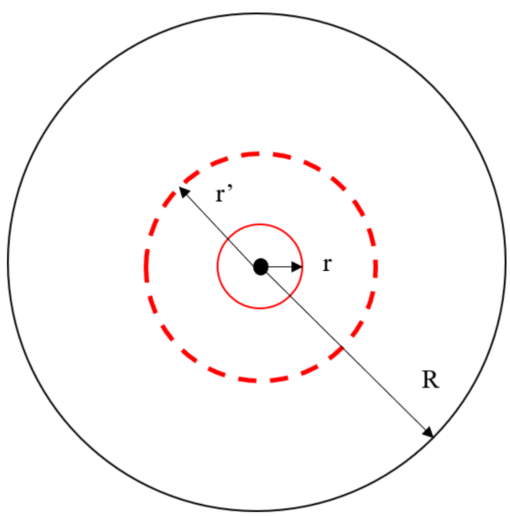

FIGURE 3.3. Illustration of collision and interaction zones

- Collision Zone (C): It is the region covered by two concentric circles of radius, r', and r. These two circles in red, represent the outer and inner boundaries of agent i respectively. The set of nodes in i's collision zone is given by

$$C_i(t) = \begin{cases} All\ j \in V : 0 < \|p_{ij}(t)\| < r' \\ empty, \quad \text{if none} \end{cases} \tag{7}$$

- Interaction Zone (I): The region between the circle of radius r' and the outermost circle of radius R is the safe zone where neighboring agents can interact using consensus laws. The set of nodes in i's interaction zone is given by:

$$I_i(t) = \begin{cases} All\ j \in V : r' < \|p_{ij}(t)\| < R \\ empty, \quad \text{if none} \end{cases} \tag{8}$$

3.3.2. Formation Generation Module

The control input applied to an individual agent is computed by the FGM. The following double integrator model is used for the leader and its $N-1$ followers:

$$\begin{cases} \dot{p}_i = v_i, \\ \dot{v}_i = u_i, \end{cases} \quad i = 1, ..., N \tag{9}$$

where, $p_i \in R^n$ is the position, $v_i \in R^n$ is the velocity and $u_i \in R^n$ is the acceleration or control input of agent, i. The network topology of the MAS is represented by a fixed undirected graph as shown in Fig. 3.2. The undirected edges represent bi-directional information exchange. In Fig. 3.2, a vertex denotes an agent, and the vertices are connected by edges. The connections of each agent i, define its neighborhood set, N_i. For instance, agent 3 has 4 neighbors given by N_3= {1, 2, 4, 5}.

The feedback control design for FGM is shown in Fig. 3.4. Inputs to the controller include data computed from the information states of neighboring agents in CN module. These are relative position, relative velocity, and the deviation from desired position state. The MAS is designed to attain a formation that is specified by the desired formation geometry as given by

$$D_r = d_{i,j} := p_{iD} - p_{jD}, \tag{10}$$

where, D_r is the relative position set, p_{iD} and p_{jD} are the desired position states of the i^{th} agent and neighboring j^{th} agent respectively. D_r is time-invariant and ensures the convergence of agents to a pre-defined formation pattern.

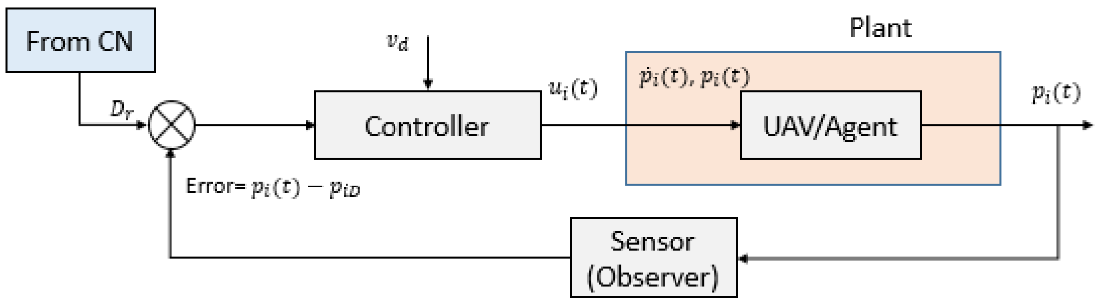

FIGURE 3.4. Formation Control Block Diagram

The control laws (input) applied to the agent i are dependent on the neighbors present in the collision zone and interaction zone. This input (u_i) includes two components as given below:

$$u_i = f_i^{con} + f_i^{rep} \tag{11}$$

where f_i^{con} is the consensus-based control force that generates the formation and preserves network connectivity. Agent i calculates this force based on the states of its logical neighbors. Repulsive force, f_i^{rep} is the second component applied to agent i to avoid collision with agents present in the collision zone, $j \in C_i$. It is based on a path planning technique, where the negative gradient of social potential function generates a repulsive force [74]. This force pulls the agents closer to the goal and pushes them away from obstacles. Here, $d_{i,j}$ is considered as the goal of an agent i whereas an agent j which may collide with the agent i is considered as an obstacle. The repulsive force starts acting in the presence of agents in the collision zone.

The consensus-based law of formation [69] applied here has been presented for a leader-follower framework, where desired velocity of leader and desired geometry of the MAS is assigned in the beginning of a mission. The control laws for the leader and the follower are given in (12) and (14) respectively:

$$f_1^{con} = f_{tr} - \sum_{j\in N_1} k_p(p_1 - p_j) + k_v(\dot{p}_1 - \dot{p}_j) + k_p \sum_{j\in N_1} d_{1j} \tag{12}$$

$$f_{tr} = -[K_p(p_1 - p_{1D}) + K_v(\dot{p}_1 - v_D)] \tag{13}$$

$$f_i^{con} = -\sum_{j\in N_i} k_p(p_i - p_j) + k_v(\dot{p}_i - \dot{p}_j) + k_p \sum_{j\in N_i} d_{ij} \tag{14}$$

where, $K_p = \gamma k_p$ and $K_v = \gamma k_v$, k_p is stiffness gain, k_v is damping gain, γ is a tracking constant, p and $\dot{p}$ are position and velocity of agents respectively, d_{ij} is the desired formation geometry for followers from $i = 2, ..., N$, v_d is the desired constant velocity given to the leader, and f_{tr} is the tracking component of the leader.

3.3.3. Collision Avoidance Module

The Collision Avoidance Module computes the repulsive force that is applied to the agent i. The block diagram of CAM is shown in Fig. 3.5.

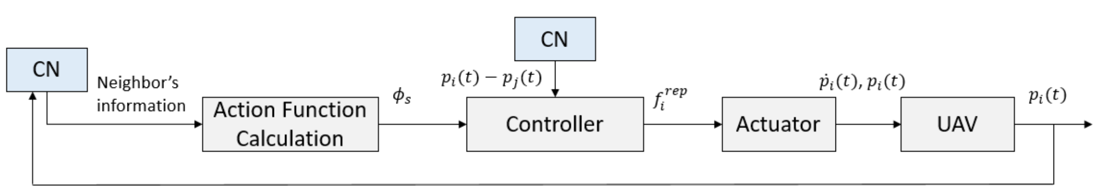

FIGURE 3.5. Collision Avoidance Module

The repulsive force f_i^{rep} in (8) is the negative gradient of repulsive potential function, $V_r(p)$. It is a smooth collective potential function defined in [75] [31] and also discussed in [70] and [71].

$$V_r(p) = \frac{1}{2}\sum_i \sum_{j \in C_i} \psi_c(\|p_j - p_i\|) \tag{15}$$

where, ψ_c is a smooth repulsive potential [73] with finite cutoff at the outer boundary of collision zone (i.e. r'). It is defined over an integral of r' to any distance (dis) value.

$$\psi_c = \int_{r'}^{dis} \phi_s(dis) \tag{16}$$

where, $dis = \|p_i - p_j\|$ and $\phi_s(dis)$ is the action function that causes the repulsive force to spring into action.

$$\phi_s(dis) = g(dis).\tilde{\phi}(dis) \tag{17}$$

Here, $\tilde{\phi}(dis) = \frac{-dis}{a+dis^2}$ and, a is design parameter. The bounds for the design parameter have been presented in [71]. The gain function $g(dis)$ is defined for the model shown in Fig. 3.2.

$$g(dis) = \begin{cases} k1, & dis \in (0, r), \\ k2, & dis \in [r, r'] \\ 0, & dis \in (r', R] \end{cases} \tag{18}$$

Here, $k2$ is defined by bump function for construction of smooth potential.

$$k2(dis) = \frac{1}{2}[1 + \cos(\pi.\frac{(dis - r)}{(r' - r)})] \tag{19}$$

where, $k1 \geq k2(dis) \geq 0$ and $k1 = 1$. This gives a repulsive force of:

$$f_i^{rep} = -\nabla V_r(p) \tag{20}$$

Using (12)-(17), we get,

$$f_i^{rep} = -\sum_{j \in C_i} \phi(\|p_i - p_j\|) \frac{p_i - p_j}{\|p_i - p_j\|} \tag{21}$$

Therefore, using equations (9)-(11) and, (18), we write the control input applied to the followers and leader in (19) and (20) respectively as:

$$u_i = -\sum_{j \in N_i} k_p(p_i - p_j) + k_v(\dot{p}_i - \dot{p}_j) + k_p \sum_{j \in N_i} d_{ij} - \sum_{j \in C_i} \phi(\|p_i - p_j\|) \frac{p_i - p_j}{\|p_i - p_j\|} \tag{22}$$

$$u_1 = f_{tr} - \sum_{j \in N_1} k_p(p_1 - p_j) + k_v(\dot{p}_1 - \dot{p}_j) + k_p \sum_{j \in N_1} d_{1j} - \sum_{j \in C_1} \phi(\|p_1 - p_j\|) \frac{p_1 - p_j}{\|p_1 - p_j\|} \tag{23}$$

The control laws in (19) and (20) will be used to generate the trajectories of agents leading to formation pattern given by desired relative position D_r.

3.4. Stability Analysis

A matrix based approach is taken for analyzing the formation control problem. Let E be the error state given by,

$$E = p - p_D, \tag{24}$$

where p and p_D are the current position and desired position state of an agent. Rearranging the consensus-based laws in (9) and (10), we get,

$$f_i^{con} = -[K_p(p_1 - p_{1D}) + K_v(\dot{p}_1 - v_D)] - \sum_{j \in N_i} [k_p(p_i - p_j) + k_v(\dot{p}_i - \dot{p}_j) - k_p d_{ij}] \tag{25}$$

where i starts from agent 1. Consensus force acting on a single agent can be written as:

$$f_i = -[\gamma k_p E + \gamma k_v \dot{E}] - \sum_{j \in N_i} (k_p E + k_v \dot{E}) \tag{26}$$

where, $K_p = \gamma k_p$ and $K_v = \gamma k_v$ are constants.

The summation operator that signifies inclusion of j neighboring agents of an agent i is reduced to the Laplacian matrix in the next equation.

$$f_i = -[\gamma k_p E + \gamma k_v \dot{E}] - [L k_p E + L k_v \dot{E}] \tag{27}$$

The Laplacian matrix denoted by L is positive semi-definite. Using the Hurwitz criterion of stability for Linear Time-Invariant Systems (LTI), we obtain a positive-definite matrix, M, by adding the Laplacian matrix, L and Pinning gain matrix, G. The first element of the Pinning gain matrix is $\gamma = 1$. The diagonal elements of the G matrix are the tracking gains of the agents as shown below. In this discussion, we consider only tracking gain for the Leader agent.

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The system dynamics reduces to:

$$f_i = -(\gamma + L) k_p E - (\gamma + L) k_v E \tag{28}$$

$$f_i = -M k_p E - M k_v E \tag{29}$$

In matrix form, the above equation can be written as:

$$\begin{bmatrix} \dot{E} \\ \ddot{E} \end{bmatrix} = \Theta * \begin{bmatrix} E \\ \dot{E} \end{bmatrix}$$

The transition matrix obtained after state space analysis of (25):

$$\Theta = \begin{bmatrix} 0 & 1 \\ -k_p M & -k_v M \end{bmatrix}$$

The eigenvalues obtained for the transition matrix, Θ are:

$$\lambda_i \pm = \frac{k_v \mu_i \pm \sqrt{k_v^2 \mu_i^2 + 4k_p * \mu_i}}{2}$$

Here, μ_i is the eigenvalue of $-M$ matrix representing the network of MAS. The feedback gains k_p and k_v should be greater than 0 for the system to be stable. When these conditions are met, all the eigenvalues have negative real parts, thus making the system stable.

3.4.1. Feedback Evaluation

The position and velocity of the agents are updated in each iteration using the motion equations, given by,

$$P_{new} = s + P_i \tag{30}$$

where,

$$s = v_i t + \frac{1}{2} u t^2 \tag{31}$$

and,

$$V_{new} = V_i t + ut \tag{32}$$

where, s is the displacement, P_{new} and V_{new} are updated position and velocity respectively. P_i and V_i are the position and velocity of agent i in current iteration, u is the acceleration calculated, and t is the time constant.

Upon updating the consensus-based acceleration with new position, P_{new} and velocity, V_{new}, we get the following control law:

$$u_i = -\sum_{j \in N_i} k_p (p_i - p_j) + (k_r' (\dot{p}_i - \dot{p}_j) + k_u' (\ddot{p}_i - \ddot{p}_j) + k_p \sum_{j \in N_i} (p_{i_D} - p_{j_D}) \tag{33}$$

where, $k_r' = k_p t + k_r$ and, $k_u' = \frac{1}{2}k_p t^2 + k_v t$. The constants, $k_r' \to k_r$ and, $k_u' \to 0$ when the time constant, t, is selected close to zero. According to the matrix analysis, the constants, k_p and k_r are chosen to be greater then 0, to ensure stability.

3.5. Results and Discussions

3.5.1. Simulation

In this paper, the consensus-based control laws are derived for formation control in multi-agent systems. The proposed model includes repulsive force based on social-potential function for collision avoidance. The repulsive force acts on each individual agent i simplifying the distributed design and minimizing information handling.

The control laws described in (19) and (20) are simulated in MATLAB to generate a predefined formation. The input values used for the simulation are as follows:

- $c1 = 5$ and, $c2 = 5$
- Desired velocity, $v_d = [7, -6]$
- Time constant, t = 0.08s
- $r = 0.4$ unit, $r' = 1$ unit and $R = 5$ units
- Design parameter for the repulsive force function, $a = 0.45$, gain value $k1 = 1$, and gain value $k2(dis)$ follows from equation (16)
- Desired geometry,

$$D_r = \begin{pmatrix} 1.1 & 1.3 \\ 1 & 0.3 \\ 3 & 0.9 \\ -7.6 & 1.6 \\ 2.5 & -4.1 \end{pmatrix}$$

For the graph topology shown in Fig. 2, the agent trajectories leading to V-shaped pattern are shown in Fig. 3.6. It takes around less than 10 iterations to reach the formation. The “x” and “o” symbols denote the initial and final locations of the agents.

In these simulations, collision avoidance is illustrated through separation distance which is observed by plotting inter-agent distance versus number of iterations required to

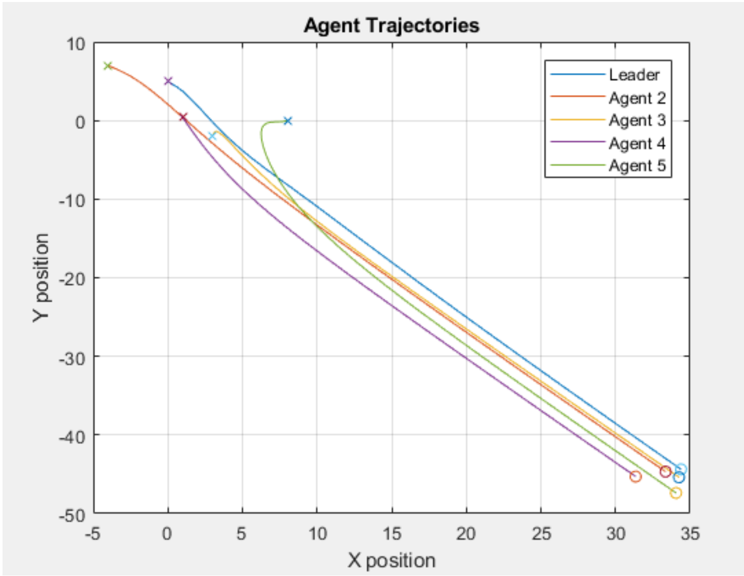

FIGURE 3.6. Agent Trajectories leading to V-shape formation

attain formation. The results are shown in Fig. 3.7 and Fig. 3.8 respectively where the separation distance between the leader and the followers are highlighted.

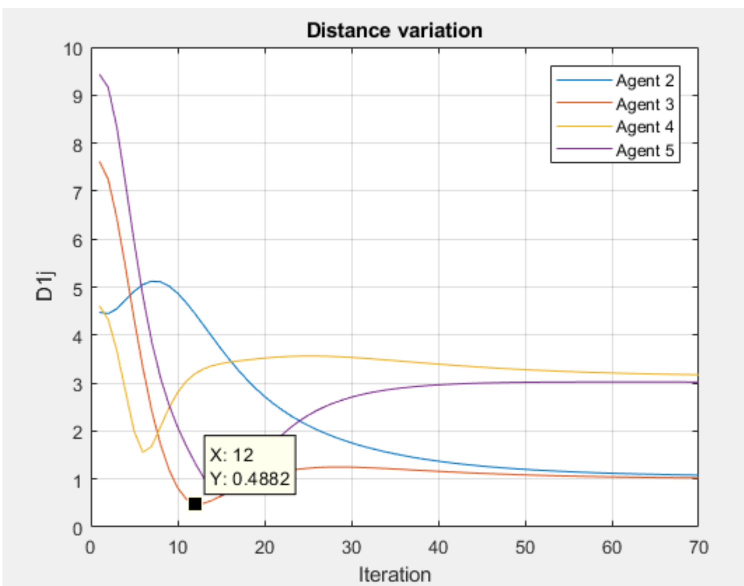

FIGURE 3.7. Separation Distance for Agent 1 without collision avoidance

The collision zone of an agent is defined by the inner and outer radius of r and r' respectively. In this case, the presence of agent 3 in collision zone of agent 1, triggers the repulsive forces. The distance between agent 1 and agent 3 (d_{13}) drops from desired $r' = 1$

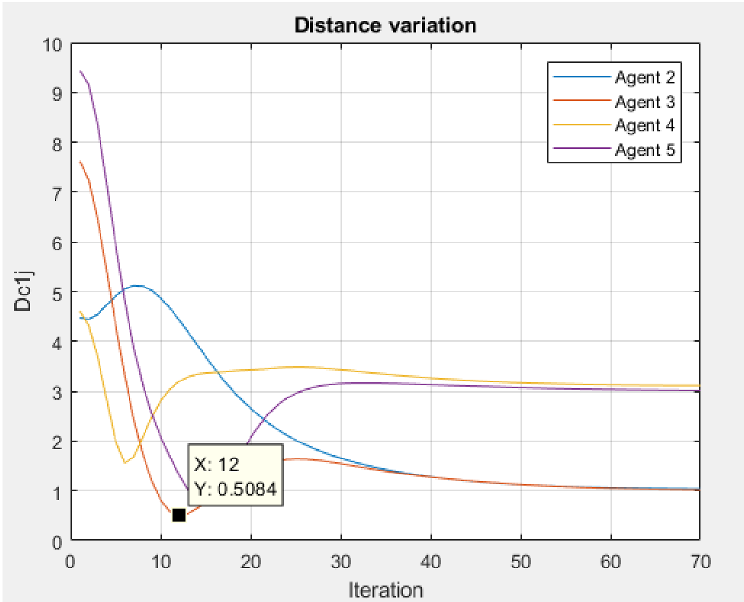

FIGURE 3.8. Separation Distance for Agent 1 with collision avoidance.

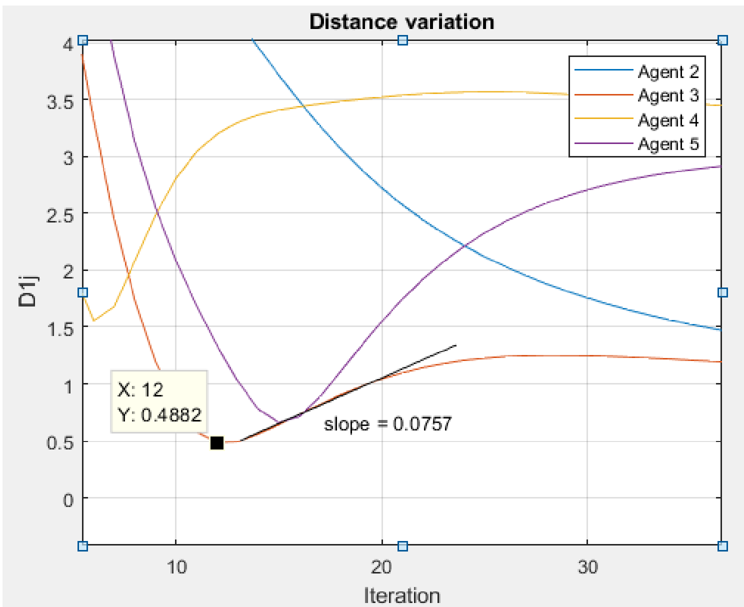

FIGURE 3.9. Slope without collision avoidance.

to 0.4882 at iteration 12. Fig. 3.8 shows the repulsive force acting on agent 1, and increasing (d_{13}) to 0.5084 at iteration 12.

A closer look of Fig. 3.7 and Fig. 3.8 is presented in Fig. 3.9 and Fig. 3.10 respectively. Acting of repulsive force in Fig. 3.10 causes a steeper slope, as compared to

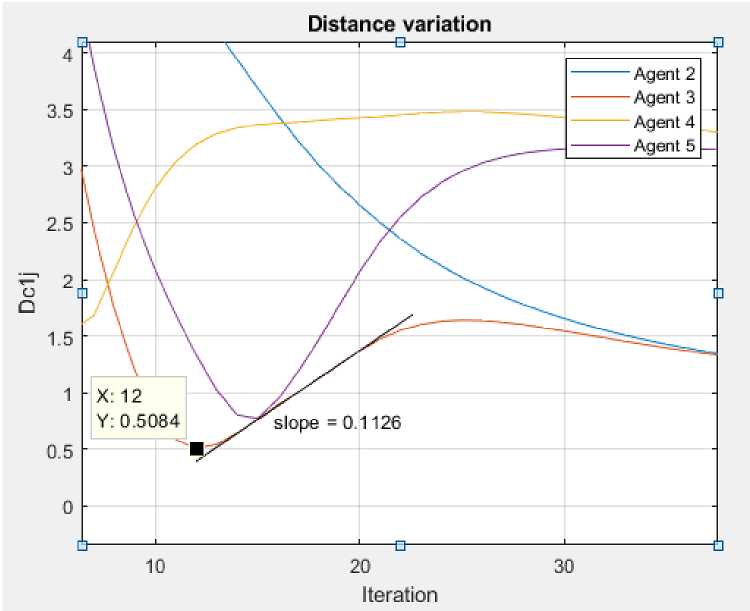

FIGURE 3.10. Slope with collision avoidance.

Fig. 3.9. It indicates a rapid increase in inter-agent distances after detecting a probable collision path among agents.

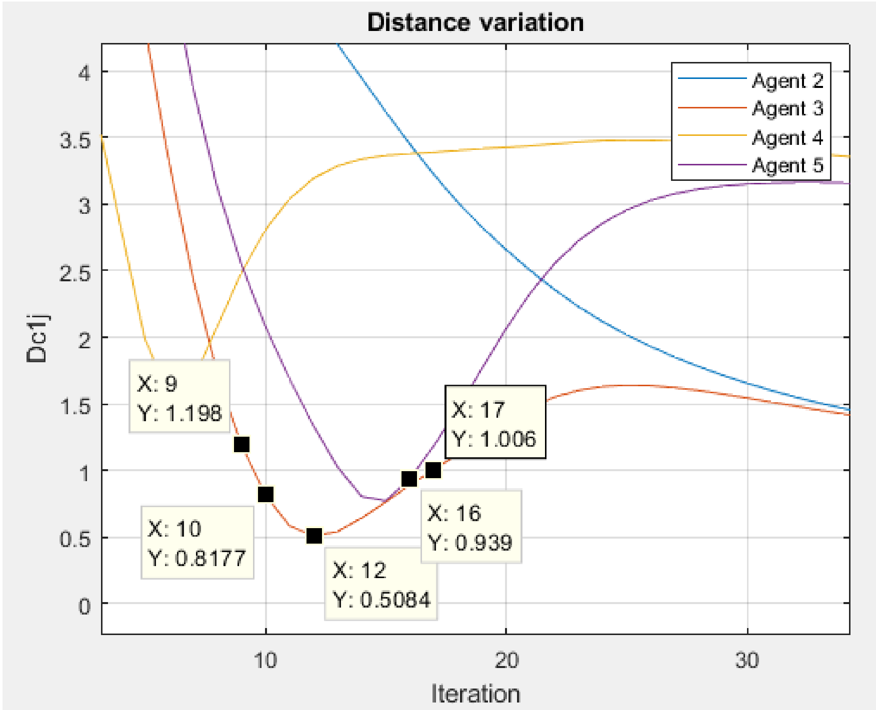

FIGURE 3.11. Activation of Repulsive force

An interesting observation made in this work is that logically connected neighbors are more prone to fall in the collision zone than other disconnected agents. Since the consensus is applied on logically connected agents, this gives rise to the situation of reaching a balance between the gains used in consensus laws and repulsive forces. Unbalanced contribution of gains will result in making one of the force components insignificant towards the computation of control input.

3.5.2. Effect of Graph Topology in Formation Control

The degree distribution of the graph topology selected for the MAS affects the formation process. The degree of an agent in MAS is the number of connections it has with other agents. In this work, degree distribution is defined as the arrangement of degrees over the nodes in a MAS. It decides whether all the agents converge to the formation at the same time, thereby affecting the convergence rate of the MAS.

The convergence to the formation by the agents is observed to be affected by the degree distribution of the graph. For a graph of N nodes, there exists a range of degree combinations by which the graph is connected. In this work, we have explored two kinds of scenarios for an undirected graph:

(1) k-regular graphs, where nodes have equal degrees

(2) Skewed graphs, where nodes have unequal degrees

The convergence performance of both k-regular and skewed graphs is compared by holding the sum of degrees of nodes as constant.

The minimum number of connections for a graph of N nodes is $N - 1$, and the maximum number of connections (complete graph) is $N(N-1)/2$. Therefore, the minimum degree per node is 1, while the maximum degree per node is $N-1$. Within this range, both k-regular and skewed graphs are considered. Let d_i be the degree of an agent i. For instance, there are two possible graph topologies such that the sum of degrees is 10.

The first topology considered for simulations is a 2-regular graph of 5 agents as shown in Fig. 3.12. Here, $d_1 = d_2 = d_3 = d_4 = d_5 = 2$.

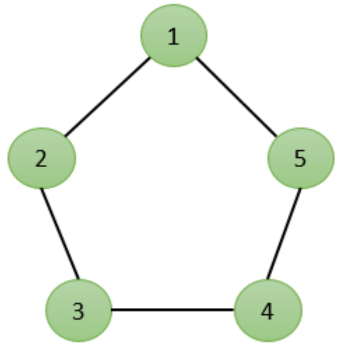

FIGURE 3.12. 2-Regular graph

The second topology shown in Fig. 3.13 is a skewed graph. In this, the sum of degrees of all the nodes add up to 10, i.e., $d_1 = d_3 = d_4 = 2$, $d_2 = 3$, and $d_5 = 1$.

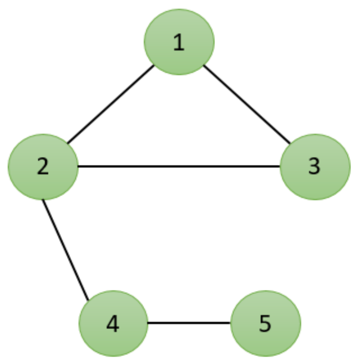

FIGURE 3.13. Skewed graph

Simulation results demonstrate that in general that regular graphs converge to desired formation faster compared to skewed graphs. In this particular simulation, the 2-regular graph took 61 iterations to reach the formation, while the skewed graph took 100 iterations to attain formation.

One way to ensure equal convergence time for all agents for the formation control process, i.e., overall convergence speed, is to have a uniform distribution of degrees among the graph vertices. This can be achieved by considering a k-regular graph. A complete graph and a k-regular graph are suitable candidates for ensuring uniform convergence.

3.6. Conclusions and Future Work

In this paper, we derived a formation control law that combines the control and connectivity constraints in a balanced fashion. The results demonstrate the expected performance of flocking and collision avoidance. The leader-follower formation structure based on consensus-based laws is applied for double integrated dynamics model. The model is extended to include social potential functions. This extension enables collision avoidance while maintaining network connectivity among agents. We have also observed the effects of the degree distribution on the convergence speed of MAS through simulation studies. The delay in convergence is found in the case of skewed graph topologies. Future work includes implementing formation with collision avoidance with switching topologies. We also intend to provide theoretical results on the impact of degree distribution on the convergence of formation control in MAS networks.

CHAPTER 4

PATH PLANNING AND TRAJECTORY TRACKING FOR HAUCS AUP PLATFORMS[1]

4.1. Introduction

In the modern era, autonomy plays an important role in all industries. Autonomous vehicles provide scope for real-time applications to be performed without human interference. Multi-Agent System (MAS) is a term used to define multiple autonomous agents working in coordination. MAS is used for modeling complex, decentralized, and real-world tasks such as package delivery by Unmanned Aircraft Systems (UAS), environmental monitoring, precision agriculture, security, disaster management, and UAS Traffic Management (UTM). One such area, where the deployment of UAS platforms could drastically improve the current labor-intensive and resource-constraint operations is aquaculture farms. In this thesis, a path planning and path following strategy for the HAUCS platform is developed. HAUCS stands for Hybrid Aerial/Underwater RobotiC System and consists of an unmanned robotic vehicle along-with submerged underwater sensors. This robotic platform has been designed to travel on the water surface and fly in the air [1] for monitoring environmental metrics. A hybrid control strategy is proposed that consists of 1) centralized path allocation to HAUCS autonomous unmanned platforms (AUP) and 2) autonomous trajectory tracking of AUPs under wind disturbances. The optimal coverage path planning is based on the Shamos Algorithm which requires the approximation of the fish farm as a convex polygon. This enables

(1) Computation of all antipodal points on the surface of polygon
(2) Finding the optimal path for each antipodal pair
(3) Selecting the path with the least number of flight lines

The proposed strategy enables periodic monitoring of mission-critical parameters in back

[1]This entire chapter is reproduced from Srijita Mukherjee, Bing Ouyang, Kamesh Namuduri, and Paul S. Wills. "Multi-Agent Systems (MAS) related data analytics in the Hybrid Aerial Underwater Robotic System (HAUCS)," in Fauzia Ahmad, Panos P. Markopoulos, and Bing Ouyang, editors, Big Data III: Learning, Analytics, and Applications, volume 11730, 2021, pp. 86-95, with permission from the International Society for Optics and Photonics (SPIE).[76]

and forth pattern (BFP). Sharing of these critical parameters among the agents allows one to estimate the state of the system and to predict potential scenarios of failures. Optimizing the area of coverage on the fish farm by each of the UASs in the presence of dynamic events will be discussed in future works.

4.2. Background

Path planning is a computational problem that deals with finding way-points for the object to move from source to destination. The navigation of robotic vehicles provides great advantages while operating in extreme scenarios. They are also expected to operate in environments filled with obstacles like a crowded building, or target recognition, etc. Therefore, robotic motion planning can be classified as path planning in a static and dynamic environment. It is easier to navigate a robot in a static environment. Graph-based representation is one of the methods to provide a safe path plan. The dynamic environment is filled with obstacles, other humans, and neighboring vehicles and requires intelligent planning techniques. Some of them are 'Ant colony optimization', 'Voronoi diagram ', 'Artificial Potential fields ', and 'Fuzzy logic methods '[77].

An important factor in the path planning of vehicles is collision avoidance. Artificial Potential Functions (APF) are a set of algorithms that are frequently used in the navigation of autonomous agents [78], [79]. The APF algorithms use the gradient of the potential function to navigate the robot. The concept of artificial potential functions is adopted from potential across opposite electric charges. The push/pull force between elements of equal/opposite charges serves as the basis of APF. The idea is that the robot moves towards a lower energy configuration. The attractive potential leads the robot to the goal, while the repulsive potential helps the robot to avoid obstacles. Applications of APF can be found in path planning [78] and task allocation [79]. APF algorithms have high scalability and low bandwidth use [80].

The Traveling Salesman Problem (TSP) is another path planning algorithm that generates an optimal path for an AUP to travel across some target locations with a minimum cost of travel. It is also known as a task allocation algorithm. It is an NP-hard problem in combinatorial optimization [1]. If the TSP problem is applied for routing vehicles with limited energy and limited weight carrying capacity, then it is characterized as a Vehicle Routing Problem (VRP). Applications of TSP/VRP algorithms are in package delivery and data collection [34]. Bio-inspired algorithms like Ant Colony Optimization are proposed in [81] for farmland monitoring.

In this thesis, we present an optimal path planning approach for the coverage of aquaculture farms. We also implement a motion control strategy for trajectory stabilization under wind conditions. The control strategy is based on sliding mode control (SMC) and adaptive control methods that deal with uncertain wind conditions.

4.2.1. Coverage Path Planning

The path planning algorithms that can be used for Hybrid Aerial Underwater Robotic System (HAUCS) AUP should be able to organize agents spatially and make decisions cooperatively [80]. Specifically, they include coverage algorithms, task allocation protocols, and motion planning algorithms. Coverage Path Planning (CPP) algorithms determine the optimal route of an AUP to cover a desired area or space while avoiding obstacles. The algorithms also consider environmental conditions such as weather changes. The agents are instructed to make continuous and sequential movements without overlapping their paths. CPP is generally used in mapping, crop monitoring, and land assessment [34]. Approaches to CPP are categorized as randomized and complete. A randomized approach does not consider the geographical information of the farm and, therefore, takes a long time to cover the whole area. On the contrary, in a complete approach, the coverage area is decomposed into cellular regions, and an optimal path is traversed to cover all the cells. Cellular decomposition and optimal path searching are few other techniques of CPP [39] [82].

A baseline algorithm based on a hybrid control system has been developed to control and regulate the HAUCS framework. The architecture of the hybrid controller is shown in Fig. 4.1. This includes a central server, operators, and individual AUPs assigned for each segment of the farm.

The goal of HAUCS mission control is to provide path plans and maintain the traffic of deployed AUPs within the context of UAS traffic management. The central server in the control architecture provides information about weather and distress calls from ponds to the operators. The server also tracks and monitors the movement of AUPs in the deployed region to make sure that their paths do not intersect. Both the central server and operators can exchange information. Each server is connected to multiple operators. The operators perform flight/path planning procedures and send instructions to the AUPs. The AUPs communicate with each other and operate in a decentralized manner. The HAUCS AUPs are assumed to be homogeneous. Each HAUCS AUP will be assigned to a home station and operate in a known environment with unpredicted variables (e.g., weather, moving obstacles, and pond conditions).

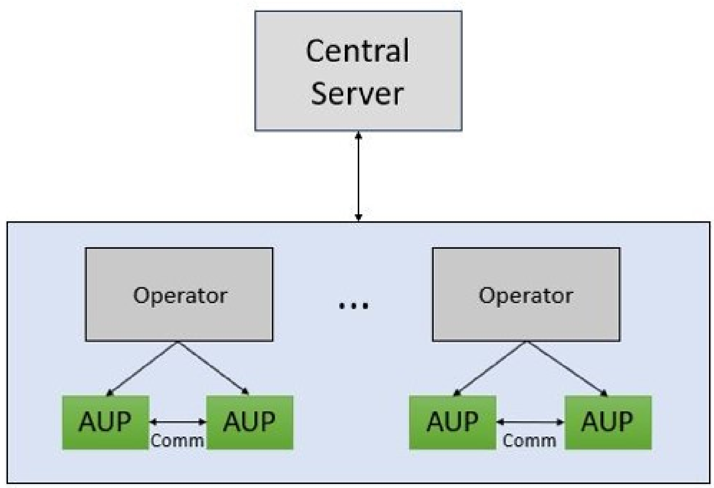

FIGURE 4.1. Hybrid Control Scheme

4.3. Problem Definition

In this section, we develop an optimal path planning strategy for covering the target area of fish farms. The fish farm is assumed to be a convex polygon, $P = (V, E)$ with the ends of the farm as vertices $V = 1, ...n$ and the sides as edges, $E = (1, 2), ..., (n, 1)$. For complete coverage of the farm, a back and forth pattern (BFP) of the flight plan is desired. Here, the AUP would take off at x_s, cover the assigned area, and land at x_e. The final flight path includes take-off and landing locations along with the way-points given by,

$W = (x_s, x_0, ..., x_m, x_e)$. The goal is to select the optimal path, which has the least number of back and forth flight lines. The first step in the strategy is to find the ideal direction of the BFP from the take-off to the landing location. The path corresponding to the minimum width of the polygon is the ideal direction for the coverage. In this context, a concept from computational geometry is found to be useful. The usage of antipodal points in deciding the best path is key to the optimal coverage of the polygon. The way-points generated by the path planner require the computation of all pairs of antipodal points on the surface of the polygon.

Algorithm 2: Optimal Path Planning Algorithm

```
Data: V, x_s, x_e
Result: W
A ← AntipodalPair(V)
c ← ∞
foreach (i, j) ∈ A do
    path ← Bestpath(V, i, j)
    if cost(x_s, path, x_e) < c then
        W ←(x_s, path, x_e)
        c ←cost(W)
    end
end
```

The algorithm then finds the best path for each antipodal pair. The best path is the one with the least back and forth flight lines, corresponding to the minimum width of the polygon. Algorithm 2 selects the path having the lowest cost, with a computational complexity of O(n).

4.3.1. Antipodal Points

The concept of antipodal points was first proposed in Shamos's Thesis [83]. Antipodal points are those vertices of a convex polygon that admit parallel lines of support such that the whole polygon lies between the two infinite parallel lines. Such vertices of a convex polygon are called antipodal pairs as shown in Fig. 4.2.

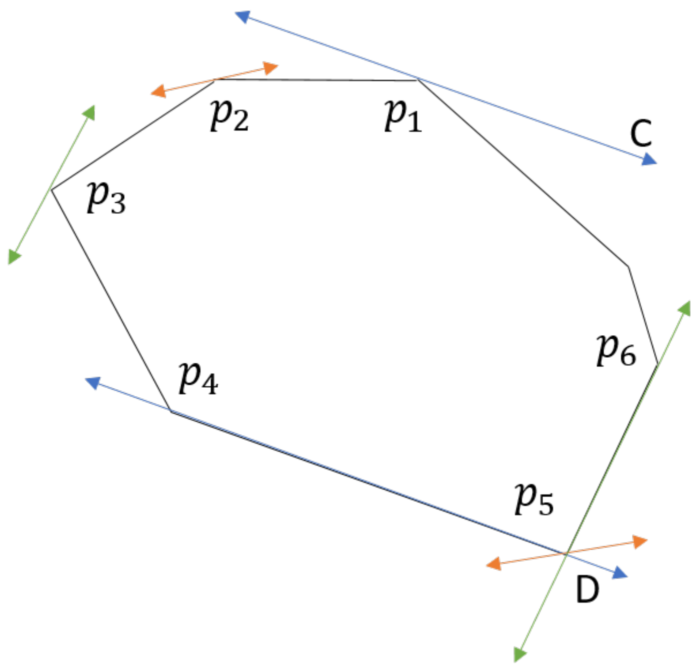

FIGURE 4.2. Antipodal Points on Convex Polygon

The parallel lines of support can either pass through a vertex or lie on an edge of the polygon. Here, lines C and D represent the parallel lines of support passing through the antipodal pair (p_1, p_4). Other pairs of antipodal points shown in the figure are (p_2, p_5), (p_3, p_6) and, (p_1, p_5). Algorithm 3 describes the Shamos algorithm to compute all the sets of antipodal points for a given convex polygon.

4.3.2. Optimal Coverage Algorithm

The antipodal pair computation gives us a set of antipodal points (i, j) for the convex polygon. The goal of the optimal coverage strategy is to find a path that traverses the minimum width of the polygon. The minimum width is traced by considering the pair of antipodal points with the shortest distance between them [84]. A block diagram of the solution approach for the whole process is shown in Fig. 4.3

The parallel lines of support are rotated on the surface of the polygon to find the best path for a given antipodal pair, as described in algorithm 4. It resembles a caliper measuring the edges of the polygon as it rotates, hence known as 'rotating caliper algorithm', as shown in Fig. 4.4. For instance, let the base edges be defined as $(b_1, b_1 + 1)$ and $(b_2, b_2 + 1)$. Firstly,

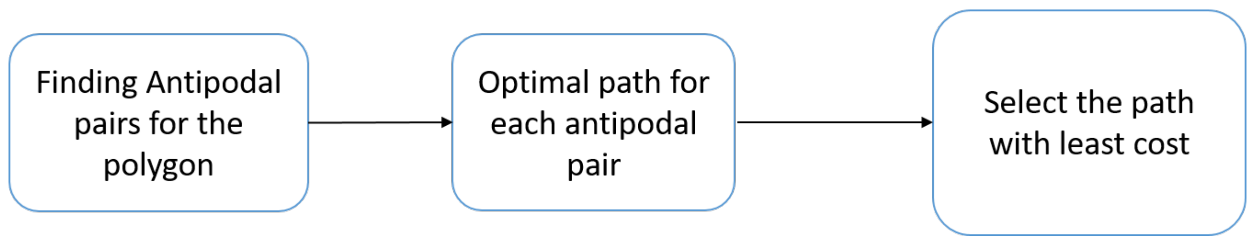

FIGURE 4.3. Sequence of Operations

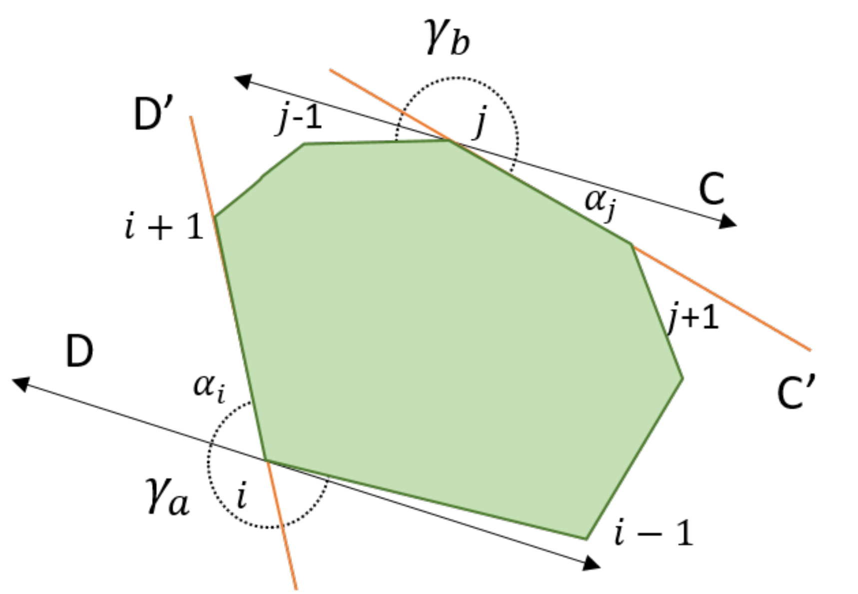

FIGURE 4.4. Rotating Caliper Illustration on Convex polygon

the lines of support (or the caliper) are rotated in the clockwise direction to find the base edge (b_1, b_1+1). Let lines C and D be the parallel lines of support passing through antipodal pair (i, j). Both the lines C and D are rotated until they touch the next adjacent vertex $(i+1)$ and $(j+1)$ respectively. The line that first touches the adjacent vertex, is assigned as the base edge of the polygon. This can be found by comparing the angles between lines of support and the adjacent vertices, α_i, and α_j. The minimum of these two angles reveals the first base edge. This gives the first path from the base edge (b_1, b_1+1) to the antipodal pair i. Secondly, the lines of support are rotated in the counterclockwise direction to reveal the second base edge of the polygon (b_2, b_2+1). To measure the angles, a line $B1$ lying on the first base (b_1, b_1+1) is considered. The angle of rotation of the lines of support from previous vertices $(i-1)$ and $(j-1)$ is compared. The lowest of the angles γ_i and γ_j gives us the second base edge. Here a path is formed from the second base (b_2, b_2+1) to the opposing vertex, j.

Algorithm 3: Shamos Algorithm

```
Data: V
Result: A
/* Find initial antipodal pair by locating vertex opposite p_1 */
i← 1 ; j← 2
while (angle(i, j) < π) do
|  j ← j + 1; current ← i; A ← (i, j)
end
/* Loop on j until whole polygon is scanned */
while j≠ n do
|  if (angle(current,i+1) < angle(current,j+1)) then
|  |  j ← j + 1; current ← j
|  else
|  |  i ← i + 1;current ← i
|  end
|  A ← (i, j)
end
/* On parallel edges */
if (angle(current,i+1) = angle(current,j+1)) then
|  A ← (i + 1, j); A ← (i, j + 1); A ← (i + 1, j + 1)
|  if current = i then
|  |  j← j + 1
|  else
|  |  i← i + 1
|  end
end
```

In this case, path 1 constitutes first base, (b_1, b_1+1) and vertex i. Path 2 starts from second base (b_2, b_2+1) until vertex j. The path with the smallest distance is selected as the best path for a given antipodal pair. Finally, after computing the best paths for all sets of antipodal points, $A = (i_1, j_1), (i_2, j_2), .., (i_n, j_n)$, the path with the lowest cost is selected for the optimal coverage of a convex polygon. The back and forth pattern (BFP) starts from the base edge and ends at the opposing antipodal point. An example of area coverage is shown in Fig. 4.5. The BFP flight lines are parallel to the lines of support of the convex polygon. The computational complexity of the given algorithm is $O(n)$.

Algorithm 4: Optimal Algorithm for Best Path

```
Data: V,(i, j)
Result: path
if angle(i, j) < angle(j, i) then
|   b₁ ← j
|   a₁ ← i
else
|   b₁ ← i
|   a₁ ← j
β ←angle(b₁, a₁) -π
γ_b₁ ←angle(b₁-1,b₁)
γ_a ←angle(a₁-1,a₁) - β
if γ_b₁ < γ_a then
|   b₂ ←(b₁ − 1)
|   a₂ ←a₁
else
|   b₂ ←(a₁ − 1)
|   a₂ ←b₁
if dist(b₁, a₁) < dist(b₂, a₂) then
|   W ←getpath(b₁, b₁ + 1)
else
|   W ←getpath(b₂ + 1, b₂)
```

4.4. AUP control strategy

In this section, the trajectory control of AUP in the presence of wind disturbances is presented [85].Initially, we assume that a central operator computes the flight plan for

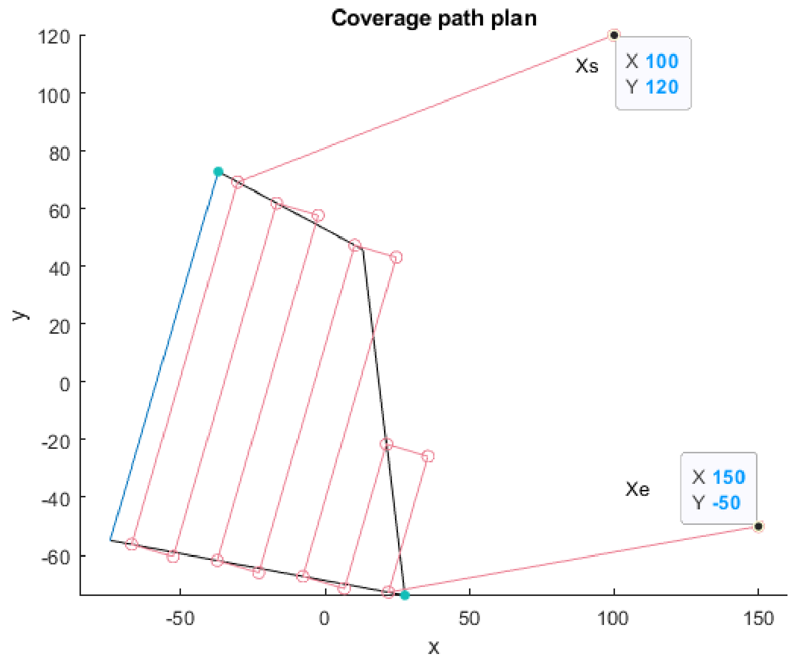

FIGURE 4.5. Generated BFP flight path

the AUPs offline. The flight path generated by the algorithm is then assigned to the AUPs ready to cover the fish farms. The path is represented as a series of way-points, $W = (x_s, x_0, ..., x_m, x_e)$, including the starting and ending location. We assume the shape of the farm as a convex polygon as shown in Fig. 4.6. The whole area is split into three sub-polygons and manually assigned to three AUPs for coverage. The way-points of the three AUPs covering the polygons in a back and forth pattern (BFP) is shown in different colors. The blue, red and, green colors display the path traversed by AUP 1, 2, and 3 respectively.

4.4.1. Kinematic Model of AUP

The kinematics of the AUP is modeled as

$$\dot{p}_i = v_i \tag{34}$$

where, the p_i is the position, and $v_i = (\dot{x}, \dot{y})$ is the translational velocity of each AUP. In the presence of wind, the disturbed kinematic model is given as

$$\dot{p}_i = v_i + W_i \tag{35}$$

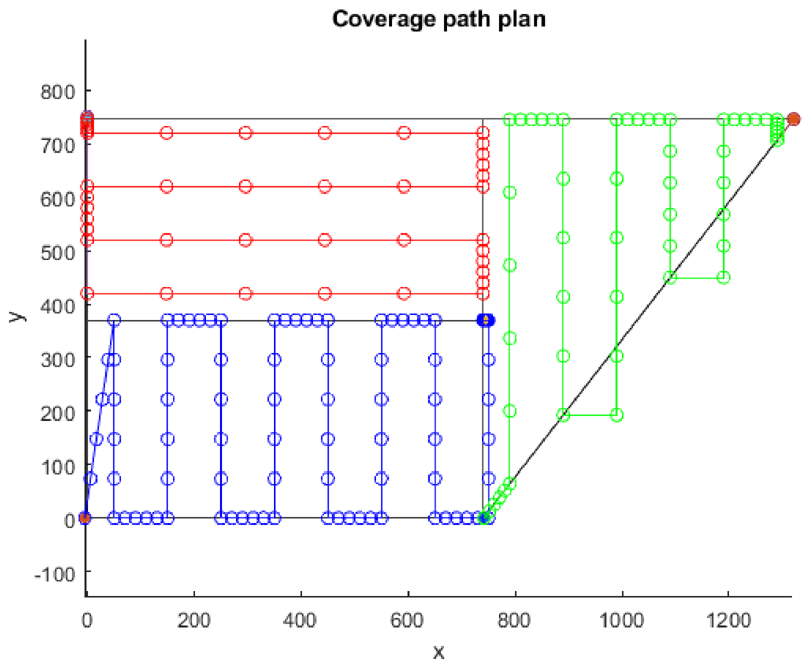

FIGURE 4.6. Coverage of Fish Farm

where, $W_i = w_0 + \Delta w(t)$, w_0 is the constant slow time-varying wind in the region, and $\Delta w(t)$ is the time-varying wind gusts. In practical scenarios, the weather in the fish farm locations may not always be available. This requires the use of adaptive control, which deals with estimated wind velocities, $\tilde{W}_0 = \hat{W}_0 - W_0$. Similarly, the wind gusts over the fish farms can also create some nonlinear uncertainties. To control the motion of AUP disturbed by wind gusts, a sliding mode control (SMC) method is used. A sliding mode control drives the system states to a sliding surface [86], where it keeps the system states in the neighborhood of the sliding surface. Normally, the error in the system states is taken as the sliding surface. A general model for a sliding surface is given by:

$$\sigma = (\frac{d}{dt} + c)^k e \tag{36}$$

where, $\sigma(x) : R^n \rightarrow R$ is a scalar function of the system state [86], c is a positive constant and, $k = n - 1$ is the relative degree between output and input variables. Substituting $k = 0$ in equation (3) gives the position error, $\tilde{e} = e - e(t)^d$ which is taken as the sliding surface.

The sliding surface, X is denoted as X1 and X2 in x and y-direction.

$$\begin{pmatrix} X1 \\ X2 \end{pmatrix} = \begin{pmatrix} \tilde{e_x} \\ \tilde{e_y} \end{pmatrix} \tag{37}$$

The control input applied to each AUP, i, for the trajectory control is given by

$$v_i = -[K_1 X + K_2 Sgn(X) + \hat{W}_0 - \dot{p^d}] \tag{38}$$

Here, K_1 and K_2 are positive-definite diagonal matrices, Sgn is the sign or signum function and, $\hat{W}_0$ is the estimated wind vector. The Lyapunov function for the system is given as:

$$V = \frac{1}{2}(X^T X + \tilde{W_0}^T A^{-1} \tilde{W}_0) \tag{39}$$

Here, A is a positive-definite gain matrix. On substituting the control input, the derivative of Lyapunov function obtained is negative semi-definite [85]. Since the path followed by AUP is time-independent, the closed-loop control system is autonomous. Therefore, the state vector of the system is asymptotically stable and the AUP converges with the desired trajectory within finite time.

4.5. Results and Discussions

In this thesis, we have presented the trajectory control of an AUP during optimal coverage of an aquaculture farm. The optimal path plan for the coverage of the farm has been implemented based on Shamos Algorithm. The usage of the rotating caliper algorithm to yield pairs of antipodal points allows computation of the most cost-effective path with a complexity of O(n). Here, the reference path following and trajectory stabilization of AUP is done autonomously. Simulation of trajectory correction in the presence of wind is performed in MATLAB as shown in Fig. 4.7. For better visualization, we have inserted 4 observation way-points between every two main way-points. The main way-points are numbered at (1,6,11,16,21,26,31,36,...), while the observation points are located in-between (2 to 5, 7 to 10, 12 to 15, 17 to 20,..).It should be noted from Fig. 4.7, that, despite the drift due to wind at main way-point 5, the AUP tracks back to the next main way-point. The observation points between main way-points 5 and 7 provide a measure of convergence to desired path plan.

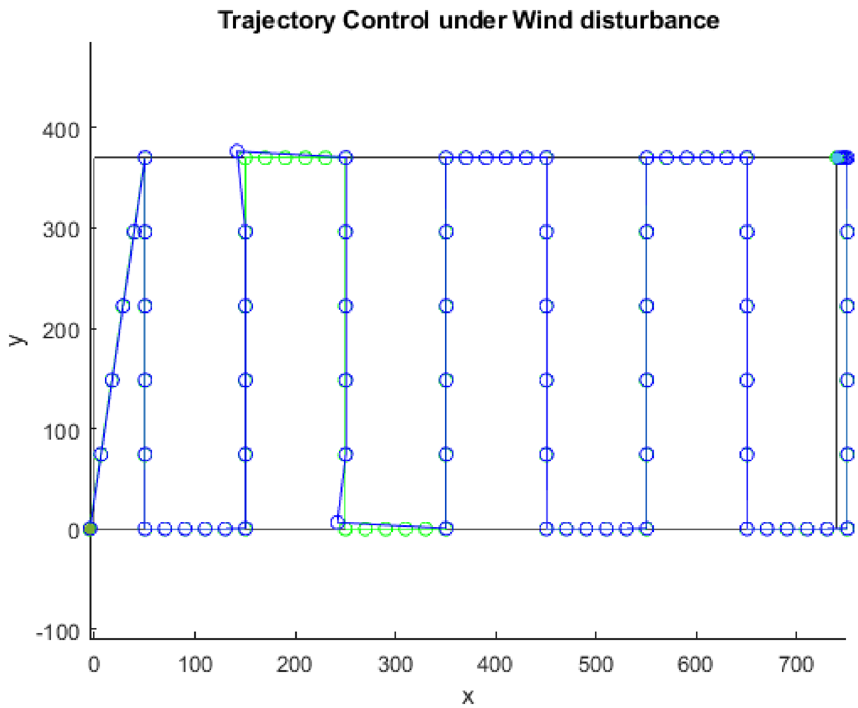

FIGURE 4.7. Trajectory Control under Wind conditions

The path re-routing by AUP when wind drift is added to main-waypoints 4 and 10, is shown in Fig. 4.8.

4.6. Conclusion

This paper presents a motion control model of HAUCS platforms for covering the aquaculture farms. Optimal path computation using the Shamos algorithm provides a back and forth pattern (BFP) of flight lines. Simulation of the trajectories of autonomous HAUCS platforms is implemented using MATLAB. The model is also tested under 2D wind conditions, and the simulation results confirm the trajectory stabilization. In the future, the later stages of the path planning will be proposed:

(1) Area assignment of AUPs to cover the fish farm
(2) Coverage planning of AUPs in a cooperative fashion

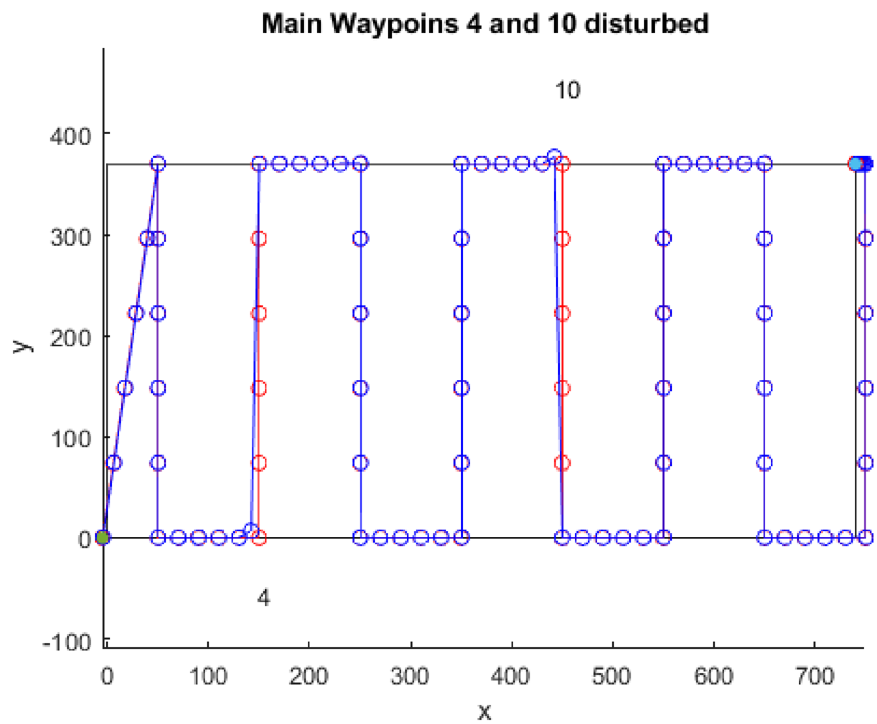

FIGURE 4.8. Trajectory tracking at main way-points 4 and 10

(3) Collision Avoidance in dynamic environment

To cope with severe weather, a control method for the third type of waypoint – protective waypoints will be presented [1]. For example, upon the detection of potential for strong winds, the control center can update the waypoints to protective waypoints to allow the HAUCS AUPs to take evasive action such as operating as USVs. The control center can restore the waypoint status at a later time when conditions return to normal.

4.7. Acknowledgments

This work was supported in part by the US Department of Agriculture, National Institute of Food and Agriculture through the National Robotics Initiative 2.0 (NRI-2.0), under Grant 2019-67022-29204

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

In this thesis, the control of multi-agent systems in the context of distributed formation control and path following in hybrid control architecture was studied.

In chapter 3, the problem of distributed formation control of multiple agents that is based on a leader-follower model is studied. The process of formation control involved two steps: (1) Formation Generation and (2) Formation Maintenance. A desired formation is represented using vectors of relative positions of neighboring agents connected by an undirected interaction graph. This undirected graph facilitates two-way communication between agents. This is decentralized system because the individual agents know the shape (or geometry) of the formation in terms of their relative distances ($D_{i,j}$). However, the final location of the formation is not known to the followers. The final formation is negotiated among the agents using consensus-based control laws.

The area around each agent is classified into various zones to characterize the different types of neighbors for each agent. The concept of logical and physical neighbors for an agent is introduced. The agents which are inter-connected in the graph topology are termed as logical agents. The information sharing during consensus takes place among the logically connected agents. The logical neighbors that do not lie in close proximity of an agent are found in the interaction zone. Whereas, physical neighbors are those that enter into the close proximity of an agent which could lead to a collision. In this work, the physical neighbors of an agent in every iteration is detected and placed them in the collision zone. Therefore, the collision zone has a time-varying collision set.

Consensus laws for a double integrator system are applied to control the multiple agents. We generate the trajectories of a V-shaped formation in MATLAB. Collision avoidance is implemented using repulsive force which is a negative gradient of the social potential function. This includes a function that activates in the presence of a physical neighbor in the collision zone. The repulsive force is applied until the agent reaches a safe distance. The safe distance is assumed as an input specification.

The effects of degree distribution of a graph topology on the convergence of the formation process is studied. The convergence performances of both k-regular and skewed graphs are compared by holding the sum of degrees of nodes as constant. It is observed that a k-regular or a complete graph leads to faster convergence compared to a skewed graph.

In chapter 4, the path following of multiple HAUCS platforms for area coverage of aquaculture farms is studied. A hybrid control architecture is implemented using the following steps: 1) the central operator assigns the task to AUPs, 2) the operator generates the path plan for the AUPs using the rotating caliper algorithm, and 3) the trajectories of autonomous HAUCS platforms are simulated using MATLAB.

The aquaculture farm is approximated as a convex polygon. The optimal path that would cover the area of the pond would be the one with least number of flight lines. The antipodal point is computed using Shamos algorithm. The possible paths between the base of the polygon and the opposite vertex is compared. The path with minimum distance is then selected as the best path for a pair of antipodal points of a convex polygon. Optimal path computation based on least cost provides a back and forth pattern (BFP) of flight lines.

For trajectory control, the HAUCS AUP is defined as a first-order kinematic system. The desired way-points generated by the operator are the reference inputs to the AUP. A trajectory tracking controller based on sliding-mode control and adaptive control is employed to track the path of the AUP under uncertain wind conditions.

Future works for this thesis includes:

(1) Formation control with collision avoidance for dynamic environments: The coordinated control of multiple vehicles in practical situations involves maneuvering in the presence of obstacles, uncertain disturbances, noise, etc. The dynamic nature of the environment also requires an online decision-making process (or a fully autonomous vehicle) to traverse the area. One of the challenges faced by a multi-agent system is the disruption in the interaction topology due to communication noise or high wind disturbances. The safest way to address such a problem is by considering a switch-

ing interaction topology of the MAS. Such a topology evolves dynamically over time and connects the nearest neighbors in the MAS for a coordinated operation.

(2) Impact of degree distribution on the convergence of formation control in MAS networks: One of the most important applications of formation control lies in area coverage or area surveillance. When a group of multiple vehicles travels in formation to retrieve mission-critical information, the rate of convergence to formation becomes a crucial factor. In this thesis, the convergence rate of the MAS is observed to be dependant on the degree distribution of the interaction topology representing the MAS. Therefore, one of the future works would be towards the theoretical confirmation of the impact of degree distribution on convergence rate in MAS applications.

(3) Cooperative control of HAUCS AUPs for area coverage of aquaculture farm: An extension of the convergence rate calculation for control of MAS in the second future work would be applicable for area coverage. The deployment of multiple HAUCS AUPs in a cooperative fashion to cover the aquaculture farms would introduce flexibility and clarity in operation.

(4) Area assignment of AUPs to cover the fish farm: The hybrid architecture developed in this thesis considers a central allocation of AUPs to cover the fish farms. However, this does not offer advantages in practical scenarios. In the real world, several constraints such as battery level, size of the fish farm, etc need to be considered before deploying the HAUCS AUPs. Hence, optimizing the number of HAUCS AUPs allocoated to cover a fish farm is an important aspect that will be carried out in the future.

(5) Collision avoidance for area coverage in a dynamic environment: The continuous and hassle-free monitoring of the aquaculture farms by HAUCS AUPs plays an important role in the HAUCS project. It is essential to introduce a distributive collision avoidance module that would navigate each HAUCS AUP among other mobile AUPs and obstacles. Local path search algorithms like artificial potential field and

behavior decomposition method and framework space approach like visibility graphs could be used for collision avoidance during area coverage.

REFERENCES

[1] B. Ouyang, P. S. Wills, Y. Tang, J. O. Hallstrom, T. C. Su, K. Namuduri, S. Mukherjee, J. I. Rodriguez-Labra, Y. Li, and C. J. Den Ouden. Initial development of the hybrid aerial underwater robotic system (HAUCS): Internet of things (IoT) for aquaculture farms. *IEEE Internet of Things Journal*, pages 1–1, 2021.

[2] W. Ren and Randal Beard. Distributed consensus in multi-vehicle cooperative control: Theory and applications. *Communications and Control Engineering*, 01 2007.

[3] Ancha. Srinivasan. *Handbook of precision agriculture : principles and applications / Ancha Srinivasan, editor*. Food Products Press New York, 2006.

[4] Subodh Bhandari, Amar Raheja, Robert L. Green, and Dat Do. Towards collaboration between unmanned aerial and ground vehicles for precision agriculture. In J. Alex Thomasson, Mac McKee, and Robert J. Moorhead, editors, *Autonomous Air and Ground Sensing Systems for Agricultural Optimization and Phenotyping II*, volume 10218, pages 26 – 39. International Society for Optics and Photonics, SPIE, 2017.

[5] F. A. Cheein, D. Herrera, J. Gimenez, R. Carelli, M. Torres-Torriti, J. R. Rosell-Polo, A. Escolà, and J. Arnó. Human-robot interaction in precision agriculture: Sharing the workspace with service units. In *2015 IEEE International Conference on Industrial Technology (ICIT)*, pages 289–295, 2015.

[6] Fishery fish farm feeder dispenses feed based on fish hunger levels. Available at `https://newatlas.com/efishery-fish-farm-smart-feeder/39940/` (2015).

[7] Mohammed M. Alammar and Ali Al-Ataby. An intelligent approach of the fish feeding system. 2018.

[8] K. R. S. R. Raju and G. H. K. Varma. Knowledge based real time monitoring system for aquaculture using iot. In *2017 IEEE 7th International Advance Computing Conference (IACC)*, pages 318–321, 2017.

[9] P. Sun and Y. Chen. Aquiculture remote monitoring system based on internet of things.

In *2019 International Conference on Robots Intelligent System (ICRIS)*, pages 187–190, 2019.

[10] Kevin L. Moore. A tutorial introduction to autonomous systems. *IFAC Proceedings Volumes*, 41(2):11720–11731, 2008. 17th IFAC World Congress.

[11] Joseph Sifakis. *Autonomous Systems – An Architectural Characterization*. 11 2018.

[12] A. Dorri, S. S. Kanhere, and R. Jurdak. Multi-agent systems: A survey. *IEEE Access*, 6:28573–28593, 2018.

[13] W. Ren, R. W. Beard, and E. M. Atkins. Information consensus in multivehicle cooperative control. *IEEE Control Systems Magazine*, 27(2):71–82, 2007.

[14] Wei Ren, R. W. Beard, and E. M. Atkins. A survey of consensus problems in multi-agent coordination. In *Proceedings of the 2005, American Control Conference, 2005.*, pages 1859–1864 vol. 3, 2005.

[15] R. Olfati-Saber, J. A. Fax, and R. M. Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.

[16] J. A. Fax and R. M. Murray. Information flow and cooperative control of vehicle formations. *IEEE Transactions on Automatic Control*, 49(9):1465–1476, 2004.

[17] Jonathan R. Lawton and Randal W. Beard. Synchronized multiple spacecraft rotations. *Automatica*, 38(8):1359–1364, 2002.

[18] Wei Ren and Randal W. Beard. Decentralized scheme for spacecraft formation flying via the virtual structure approach. *Journal of Guidance, Control, and Dynamics*, 27(1):73–82, 2004.

[19] J. Lin, A. S. Morse, and B. D. O. Anderson. The multi-agent rendezvous problem. In *42nd IEEE International Conference on Decision and Control (IEEE Cat. No.03CH37475)*, volume 2, pages 1508–1513 Vol.2, 2003.

[20] J. Cortes, S. Martinez, and F. Bullo. Robust rendezvous for mobile autonomous agents via proximity graphs in arbitrary dimensions. *IEEE Transactions on Automatic Control*, 51(8):1289–1298, 2006.

[21] R. Olfati-Saber. Flocking for multi-agent dynamic systems: algorithms and theory. *IEEE Transactions on Automatic Control*, 51(3):401–420, 2006.

[22] Yang Quan Chen and Zhongmin Wang. Formation control: a review and a new consideration. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3181–3186, 2005.

[23] R. Olfati-Saber, J. A. Fax, and R. M. Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.

[24] Yongcan Cao, Wenwu Yu, Wei Ren, and Guanrong Chen. An overview of recent progress in the study of distributed multi-agent coordination. *IEEE Transactions on Industrial Informatics*, 9, 07 2012.

[25] T. Balch and R. C. Arkin. Behavior-based formation control for multirobot teams. *IEEE Transactions on Robotics and Automation*, 14(6):926–939, 1998.

[26] Kar-Han Tan and M. A. Lewis. Virtual structures for high-precision cooperative mobile robotic control. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS '96*, volume 1, pages 132–139 vol.1, 1996.

[27] Kwang-Kyo Oh, Myoung-Chul Park, and Hyo-Sung Ahn. A survey of multi-agent formation control. *Automatica*, 53:424–440, 2015.

[28] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The internet of things: A survey. *Computer Networks*, 54(15):2787–2805, 2010.

[29] O. Elijah, T. A. Rahman, I. Orikumhi, C. Y. Leow, and M. N. Hindia. An overview of internet of things (iot) and data analytics in agriculture: Benefits and challenges. *IEEE Internet of Things Journal*, 5(5):3758–3773, 2018.

[30] 8 digital technologies disrupting aquaculture. Available at `https://www.alltech.com/blog/8-digital-technologies-disrupting-aquaculture` (2017).

[31] Hakan Yavuz and A. Bradshaw. A new conceptual approach to the design of hybrid control architecture for autonomous mobile robots. *Journal of Intelligent and Robotic Systems*, 34:1–26, 05 2002.

[32] Robotic paradigm. Available at `https://en.wikipedia.org/wiki/Robotic_paradigm` (2020).

[33] Bhushan Mahajan and Punam Marbate. Literature review on path planning in dynamic environment. 2013.

[34] J. Xie, L. R. Garcia Carrillo, and L. Jin. Path planning for uav to cover multiple separated convex polygonal regions. *IEEE Access*, 8:51770–51785, 2020.

[35] Daniel E Koditschek and Elon Rimon. Robot navigation functions on manifolds with boundary. *Advances in Applied Mathematics*, 11(4):412–442, 1990.

[36] T. Weigel, J. . Gutmann, M. Dietl, A. Kleiner, and B. Nebel. Cs freiburg: coordinating robots for successful soccer playing. *IEEE Transactions on Robotics and Automation*, 18(5):685–699, 2002.

[37] Federico Rossi, Saptarshi Bandyopadhyay, Michael Wolf, and Marco Pavone. Review of multi-agent algorithms for collective behavior: a structural taxonomy. *IFAC-PapersOnLine*, 51(12):112–117, 2018. IFAC Workshop on Networked & Autonomous Air & Space Systems NAASS 2018.

[38] Enric Galceran and Marc Carreras. A survey on coverage path planning for robotics. *Robotics and Autonomous Systems*, 61(12):1258–1276, 2013.

[39] Enric Galceran and Marc Carreras. A survey on coverage path planning for robotics. *Robotics and Autonomous Systems*, 61(12):1258–1276, 2013.

[40] Mia Stevens and Ella Atkins. Geofencing in immediate reaches airspace for unmanned aircraft system traffic management. 01 2018.

[41] Y. Liu, R. Lv, X. Guan, and J. Zeng. Path planning for unmanned aerial vehicle under geo-fencing and minimum safe separation constraints. In *2016 12th World Congress on Intelligent Control and Automation (WCICA)*, pages 28–31, 2016.

[42] J. Yang, X. Wang, Z. Li, P. Yang, X. Luo, K. Zhang, S. Zhang, and L. Chen. Path planning of unmanned aerial vehicles for farmland information monitoring based on wsn. In *2016 12th World Congress on Intelligent Control and Automation (WCICA)*, pages 2834–2838, 2016.

[43] D. Calzolari, B. Schürmann, and M. Althoff. Comparison of trajectory tracking controllers for autonomous vehicles. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–8, 2017.

[44] R. DeCarlo and S. Żak. A quick introduction to sliding mode control and its applications 1. 2008.

[45] Andrew Zulu and Samuel John. A review of control algorithms for autonomous quadrotors. *Open Journal of Applied Sciences*, 04:547–556, 01 2014.

[46] Sundarapandian Vaidyanathan and Ahmad Taher Azar. Chapter 1 - an introduction to backstepping control. In Sundarapandian Vaidyanathan and Ahmad Taher Azar, editors, *Backstepping Control of Nonlinear Dynamical Systems*, Advances in Nonlinear Dynamics and Chaos (ANDC), pages 1–32. Academic Press, 2021.

[47] Craig W Reynolds. Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 25–34, 1987.

[48] Fabrizio Giulietti and Giovanni Mengali. Dynamics and control of different aircraft formation structures. *Aeronautical Journal*, 108(1081):117–124, 2004.

[49] Y. Chen, J. Lu, X. Yu, and D. J. Hill. Multi-agent systems with dynamical topologies: Consensus and applications. *IEEE Circuits and Systems Magazine*, 13(3):21–34, 2013.

[50] Krishna S Raghuwaiya, Shonal Singh, and Jito Vanualailai. Formation control of mobile robots. *International Journal of Mechanical and Mechatronics Engineering*, 5(12):2697–2702, 2011.

[51] T. Balch and R. C. Arkin. Behavior-based formation control for multirobot teams. *IEEE Transactions on Robotics and Automation*, 14(6):926–939, 1998.

[52] A. Jadbabaie, Jie Lin, and A. S. Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48(6):988–1001, 2003.

[53] G. Antonelli, F. Arrichiello, and S. Chiaverini. Experiments of formation control with

multirobot systems using the null-space-based behavioral control. *IEEE Transactions on Control Systems Technology*, 17(5):1173–1182, 2009.

[54] Wei Ren and Randal W. Beard. Decentralized scheme for spacecraft formation flying via the virtual structure approach. *Journal of Guidance, Control, and Dynamics*, 27(1):73–82, 2004.

[55] T. H. A. van den Broek, N. van de Wouw, and H. Nijmeijer. Formation control of unicycle mobile robots: a virtual structure approach. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 8328–8333, 2009.

[56] J. P. Desai, J. P. Ostrowski, and V. Kumar. Modeling and control of formations of nonholonomic mobile robots. *IEEE Transactions on Robotics and Automation*, 17(6):905–908, 2001.

[57] Jaydev P. Desai. A graph theoretic approach for modeling mobile robot team formations. *Journal of Robotic Systems*, 19(11):511–525, 2002.

[58] Luca Consolini, Fabio Morbidi, Domenico Prattichizzo, and Mario Tosques. Leader–follower formation control of nonholonomic mobile robots with input constraints. *Automatica*, 44(5):1343–1349, 2008.

[59] Jian Chen, Dong Sun, Jie Yang, and Haoyao Chen. Leader-follower formation control of multiple non-holonomic mobile robots incorporating a receding-horizon scheme. *I. J. Robotic Res.*, 29:727–747, 05 2010.

[60] Dimos V. Dimarogonas, Panagiotis Tsiotras, and Kostas J. Kyriakopoulos. Leader–follower cooperative attitude control of multiple rigid bodies. *Systems & Control Letters*, 58(6):429–435, 2009.

[61] Hesheng Wang, Dejun Guo, Xinwu Liang, Weidong Chen, Guoqiang Hu, and Kam K Leang. Adaptive vision-based leader–follower formation control of mobile robots. *IEEE Transactions on Industrial Electronics*, 64(4):2893–2902, 2016.

[62] Falin Wu, Jiemin Chen, and Yuan Liang. Leader-follower formation control for quadrotors. In *IOP Conf. Ser. Mater. Sci. Eng*, volume 187, page 012016, 2017.

[63] Xiangke Wang, Zhiwen Zeng, and Yirui Cong. Multi-agent distributed coordination control: Developments and directions via graph viewpoint. *Neurocomputing*, 199:204–218, 2016.

[64] Tingrui Han, Zhiyun Lin, Ronghao Zheng, and Minyue Fu. A barycentric coordinate-based approach to formation control under directed and switching sensing graphs. *IEEE Transactions on cybernetics*, 48(4):1202–1215, 2017.

[65] W. Ren. On consensus algorithms for double-integrator dynamics. *IEEE Transactions on Automatic Control*, 53(6):1503–1509, 2008.

[66] B. D. O. Anderson, C. Yu, B. Fidan, and J. M. Hendrickx. Rigid graph control architectures for autonomous formations. *IEEE Control Systems Magazine*, 28(6):48–63, 2008.

[67] S. Mukherjee and K. Namuduri. Joint flocking and deconfliction in unmanned aerial vehicle swarms. In *MILCOM 2019 - 2019 IEEE Military Communications Conference (MILCOM)*, pages 49–55, 2019.

[68] W. Ren, R. W. Beard, and E. M. Atkins. Information consensus in multivehicle cooperative control. *IEEE Control Systems Magazine*, 27(2):71–82, 2007.

[69] Suresh M Joshi and Oscar R Gonzalez. Consensus-based formation control of a class of multi-agent systems. 2014.

[70] Arindam Mondal and Laxmidhar Behera. Gradient-based collision free desired formation generation. *IFAC Proceedings Volumes*, 47(1):448–454, 2014.

[71] Arindam Mondal, Laxmidhar Behera, Soumya Ranjan Sahoo, and Anupam Shukla. A novel multi-agent formation control law with collision avoidance. *IEEE/CAA Journal of Automatica Sinica*, 4(3):558–568, 2017.

[72] Zhiyong Chen, Ming-Can Fan, and Hai-Tao Zhang. How much control is enough for network connectivity preservation and collision avoidance? *IEEE transactions on cybernetics*, 45(8):1647–1656, 2014.

[73] Xiaoyuan Luo, Shaobao Li, and Xinping Guan. Flocking algorithm with multi-target tracking for multi-agent systems. *Pattern Recognition Letters*, 31(9):800–805, 2010.

[74] Kevin M Lynch and Frank C Park. *Modern Robotics*. Cambridge University Press, 2017.

[75] Reza Olfati-Saber. Flocking for multi-agent dynamic systems: Algorithms and theory. *IEEE Transactions on automatic control*, 51(3):401–420, 2006.

[76] Srijita Mukherjee, Bing Ouyang, Kamesh Namuduri, and Paul S. Wills. Multi-Agent Systems (MAS) related data analytics in the Hybrid Aerial Underwater Robotic System (HAUCS). In Fauzia Ahmad, Panos P. Markopoulos, and Bing Ouyang, editors, *Big Data III: Learning, Analytics, and Applications*, volume 11730, pages 86 – 95. International Society for Optics and Photonics, SPIE, 2021.

[77] Bhushan Mahajan and Punam Marbate. Literature review on path planning in dynamic environment. 2013.

[78] Daniel E Koditschek and Elon Rimon. Robot navigation functions on manifolds with boundary. *Advances in Applied Mathematics*, 11(4):412–442, 1990.

[79] T. Weigel, J. . Gutmann, M. Dietl, A. Kleiner, and B. Nebel. Cs freiburg: coordinating robots for successful soccer playing. *IEEE Transactions on Robotics and Automation*, 18(5):685–699, 2002.

[80] Federico Rossi, Saptarshi Bandyopadhyay, Michael Wolf, and Marco Pavone. Review of multi-agent algorithms for collective behavior: a structural taxonomy. *IFAC-PapersOnLine*, 51(12):112–117, 2018. IFAC Workshop on Networked & Autonomous Air & Space Systems NAASS 2018.

[81] J. Yang, X. Wang, Z. Li, P. Yang, X. Luo, K. Zhang, S. Zhang, and L. Chen. Path planning of unmanned aerial vehicles for farmland information monitoring based on wsn. In *2016 12th World Congress on Intelligent Control and Automation (WCICA)*, pages 2834–2838, 2016.

[82] Howie Choset. Coverage for robotics - a survey of recent results. *Ann. Math. Artif. Intell.*, 31:113–126, 10 2001.

[83] Michael lail Shamos. *Computational Geometry*. PhD dissertation, Yale University, 1978.

[84] J. I. Vasquez Gomez, M. M. Melchor, and J. C. Herrera Lozada. Optimal coverage path planning based on the rotating calipers algorithm. In *2017 International Conference*

on Mechatronics, Electronics and Automotive Engineering (ICMEAE), pages 140–144, 2017.

[85] Juan Escareno, Sergio Salazar, Hugo Romero, and R. Lozano. Trajectory control of a quadrotor subject to 2d wind disturbances. *Journal of Intelligent & Robotic Systems*, 70, 04 2012.

[86] R. DeCarlo and S. Zak. A quick introduction to sliding mode control and its applications 1. 2008.